Teaching AI to Listen in a Multilingual World: From Single to Multi-Sequence Speech Recognition

Brian Yan

CMU-LTI-26-007

April 2026

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Shinji Watanabe **(chair)**, Carnegie Mellon University
David R. Mortensen, Carnegie Mellon University
Graham Neubig, Carnegie Mellon University
Matthew Wiesner, Johns Hopkins University

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Language and Information Technology.

Keywords: Multilingual Speech Recognition, Code-Switched Speech Recognition, Language Identification, Language Diarization, Speech-to-Speech Translation, Real-Time Speech Translation, Multi-Sequence Modeling

To my family

Abstract

In the neural network era, sequence modeling frameworks for speech-to-text processing conventionally operate by directly mapping a spoken input to a *single* textual output sequence. On the canonical task of transcribing monolingual speech, these types of systems have already achieved human-like accuracy on highly resourced languages. However, this single-sequence paradigm is fundamentally limited when it comes to more multilingual speech processing tasks that necessitate modeling *multiple* textual sequences from a single spoken input.

This thesis explores two sub-types of multi-sequence modeling: (1) *parallel*, where multiple plausible textual sequences are considered jointly and (2) *sequential*, where multiple textual sequences are produced in a conditional chain.

Parallel multi-sequence modeling is required for a class of problems where speech-to-text mappings are dependent on a classification task. We start from multilingual speech recognition in the wild, where spoken language identity is unknown and to be predicted by the system. We then move to address the scenario that spoken language identity is variable within an utterance, as in code-switched speech produced by bilingual speakers.

Sequential multi-sequence modeling is required for a class of problems where the overall task consists of multiple sub-tasks, such as speech translation which consists of speech recognition and machine translation. We start from the offline speech translation setting, where systems process pre-recorded speech. We then move to address the streaming setting, where transcript and translation texts are output with low latency while speech is still being continuously produced.

The techniques developed in this thesis are first validated by building dedicated systems for particular use cases, including nuanced areas of human language such as bilingualism and dialectal diglossia. We then propose to apply these techniques to adapt existing large-scale foundation models for more effective processing of massively multilingual speech.

Acknowledgment

First, I would like to express my deepest gratitude to my advisor, Shinji Watanabe, for their guidance, support, and patience throughout my PhD. I will always remember our chats in the hallway of the 6th floor of the GHC building. Thank you for showing me the joys of research. And most of all thank you for allowing me to learn and grow as your student.

I am also grateful to my committee members, Matthew Wiesner, David Mortensen, and Graham Neubig, for their support throughout my PhD. Matthew, you became a great friend and mentor to me. Surely, I couldn't have done it without you. David and Graham, you inspired me when I first came to the LTI and working together was my honor.

I was also fortunate to collaborate with a great number of amazing people: Siddharth Dalmia, Xuankai Chang, Siddhant Arora, Yifan Peng, Jiatong Shi, William Chen, Jinchuan Tian, Patrick Fernandes, Shikhar Bharadwaj, Siqi Ouyang, Samuele Cornell, Sanjeev Khudanphur, Peter Viechnicki, Ahmed Ali, Michael Auli, Chunlei Zhang, Injy Hamed, Alexander Polok, Puyuan Peng, and so many more.

To my friends and family, thank you for always supporting me. Iris and Koshak, thank you for being my everything.

Contents

Chapter 1

Introduction

1.1 Background and Motivation

The Big Picture

Speech processing applications are evolving, becoming increasingly multilingual year over year. On one axis, systems are becoming wider in their language coverage, progressing from a handful of languages to hundreds and thousands of languages (Radford et al., 2023; Pratap et al., 2024; Li et al., 2022b). This is often achieved within a single system, commonly referred to as foundation models (Bommasani et al., 2021), which are general purpose models trained on large quantities of diverse data across many languages. These systems are also becoming wider in the sense that they are performing multiple tasks; rather than just performing transcription in the source language, there is a progression towards performing transcription as well as translation (Radford et al., 2023; Rubenstein et al., 2023; Seamless Communication, 2023).

On the second axis, systems are also becoming deeper in their language coverage. Rather than only broadly covering the standard forms of languages, dedicated systems are supporting increasingly specific linguistic communities. For instance, there are efforts towards increasing the coverage of the nuances of human language, such as regional dialects (Agarwal et al., 2023) and code-switching between languages (Shi et al., 2020; Diwan et al., 2021a).

This thesis explores the underlying sequence modeling frameworks for multilingual speech processing tasks. The canonical monolingual sequence modeling approach in the neural network era is a single sequence prediction paradigm: a single sequence input (speech) yields a single sequence output (text). The popular neural network frameworks, such as the Attentional Encoder Decoder (AED) (Chan et al., 2016), Connectionist Temporal Classification (CTC) (Graves et al., 2006), and Transducer (Graves, 2012a) all generally operate in this manner. However, as applications become increasingly multilingual systems also have taken on additional requirements which

necessitate modifications to this single sequence prediction status quo.

We focus on two particular items: 1) systems now need to **determine what language is being spoken** and 2) systems now have to **perform both transcription and translation**.

Motivation: Beyond Single-Sequence Modeling

Modern speech recognition systems are built around a single-sequence paradigm: a speech signal is mapped directly to a single textual output. This formulation underlies widely used approaches such as CTC, encoder-decoder, and transducer models, and has been highly effective in monolingual settings. In multilingual speech, however, this formulation becomes limiting. The language of the input is not known a priori and must be inferred from the signal itself. At the same time, speech recognition is inherently streaming and causal: the model processes audio incrementally and must make predictions before the full utterance is observed.

This leads to a mismatch: the model must make decisions about language and interpretation from partial input, even when the evidence is not yet sufficient. Therefore, we consider:

- **Language identification and recognition are entangled (massively multilingual speech).** In large-scale multilingual settings (e.g., 100+ languages), the language of the input is not known and must be inferred jointly with recognition. Standard approaches rely on a single predicted language, but errors in this prediction directly affect the decoded text. In our massively multilingual experiments, we observe that this early commitment limits the effectiveness of language-conditioned models. This motivates maintaining multiple language-conditioned hypotheses in parallel, rather than committing to a single language upfront (Part I).

- **Language varies over time (code-switching).** In code-switched speech, the language is not fixed but changes within an utterance. This occurs both in intra-sentential switching (within a sentence) and inter-sentential switching (across utterances). While inter-sentential switching can often be handled by segmenting and applying standard multilingual LID + ASR pipelines, intra-sentential switching requires fine-grained, time-resolved language inference. Our work on CS-FLEURS, CS-YODAS, and models such as LaDiCoW explicitly addresses this setting by modeling language as a time-varying signal and composing outputs across languages (Part II).

- **Multiple output sequences (transcription and translation).** In many applications, the goal is not only to transcribe speech but also to translate it. This is particularly important in dialectal settings, where the spoken form may not have a standardized or well-resourced written form. In such cases, the transcription alone may be noisy or ambiguous, and the translation provides an additional signal for interpretation. This results in two correlated

output sequences, a transcription and a translation, that must be generated and reconciled jointly. Our work on multilingual and dialectal speech translation highlights this setting, where a single-sequence formulation is insufficient to capture the relationship between these outputs (Part III).

- **Simultaneous and streaming systems (online inference).** In streaming ASR and speech-to-speech translation, systems must produce outputs incrementally as audio is observed. This requires making decisions under latency constraints, often before sufficient context is available. In practice, this leads to pipelines that generate intermediate sequences (e.g., partial transcripts, intermediate translations) that are refined over time. These systems naturally operate over multiple evolving sequences, rather than a single fixed output, further motivating a multi-sequence formulation (Part IV).

These settings share a common structure: the system must reason over multiple plausible interpretations of the inpu, across languages, across time, and across tasks, rather than commit to a single sequence early in the inference process. The single-sequence paradigm forces these decisions to be made implicitly and prematurely, which limits performance in multilingual and streaming settings.

This thesis adopts a different perspective. Instead of collapsing uncertainty into a single output, we explicitly model and maintain multiple sequences during inference. Each sequence corresponds to a different language-conditioned or task-conditioned interpretation, and the model defers commitment until sufficient evidence is available.

Under this view, multilingual speech processing is more naturally framed as a *multi-sequence* problem, where the goal is not to predict a single sequence, but to generate, align, and reconcile multiple candidate sequences over time. The following sections formalize this perspective and develop methods that operationalize it across the settings described above.

The Canonical Approach

To accommodate these additional requirements, practitioners have initially opted for minor modifications to the canonical monolingual approach. For instance, the Whisper model (Radford et al., 2023), is an attentional encoder decoder with the following two changes. In figure 1.1, we illustrate the computational form of these approaches.

First, in order to determine what language is being spoken, the model first makes a classification decision and then uses that language as a mode for outputting text - this is still a single sequence prediction. Then to perform both transcription and translation in the same model, a task mode was introduced. This is a toggle between either outputting transcription text or translation text - again, still a single sequence prediction.

Figure 1.1: Single Sequence Modeling

Re-formulation: Single to Multiple Sequence Prediction

To demonstrate the need for multi-sequence prediction, we start by revisiting the probabilistic formulation of sequence prediction. We'll then address how the formulation changes to accommodate the two aforementioned multilingual system requirements of 1) determining what language is being spoken and 2) performing both transcription and translation.

Single Sequence Prediction

The canonical monolingual speech recognition task is to find Y which maximizes the conditional likelihood of Y given X, where $Y = (y_m \in \mathcal{V} | m = 1, ..., M)$ is a M-length text sequence consisting of tokens from a vocabulary $\mathcal{V}$, and $X = (\mathbf{x}_t \in R^D | t = 1, ..., T)$ is a T-length speech sequence of D-dimensional space:

$$p(Y|X), \tag{1.1}$$

which is a single sequence prediction framework - multilingual speech processing requires more.

Find Y^* which maximizes $p(Y|X)$, or
Find Y'^* which maximizes $p(Y'|X)$

Multiple Sequence Prediction

Now to additionally 1) determine what language is being spoken, the formulation changes to consider maximizing the joint likelihood between the language identity, l, and the text in that language, Y. This joint likelihood can be broken down by product rule:

$$p(l, Y|X) = p(l|X)p(Y|l, X), \tag{1.2}$$

making it possible to model this task by first choosing the language l and then predicting text in that language Y. Notice here that this chained computation is dependent on an initial classification

problem, spoken language identification, represented by $p(l, X)$. Given that the set of known languages is bounded, it is possible to *parallelize* the modeling of the dependent task, transcription, represented by $p(Y|l, X)$, by predicting text across multiple language modes at the same time.

In the other case, to additionally 2) perform both transcription and translation, the formulation first changes similarly to consider a joint likelihood; this time between the transcription text, Y, and the translation text, Y'. But there is a critical difference. This time the additional task is not a classification, but rather another sequence prediction:

$$p(Y, Y'|X) = p(Y|X)p(Y'|X, Y), \tag{1.3}$$

meaning we must first model the prediction of transcription text, Y, before we are able to start predicting translation text, Y' - this chained computation is dependent on an initial sequence problem, source language transcription, represented by $p(Y|X)$. The search space of all possible transcription sequences, Y, unlike that of all possible languages, l, is unbounded and not known in advance of receiving the speech information, X. It is therefore not possible to parallelize the dependent task, translation, represented by $p(Y'|X, Y)$,

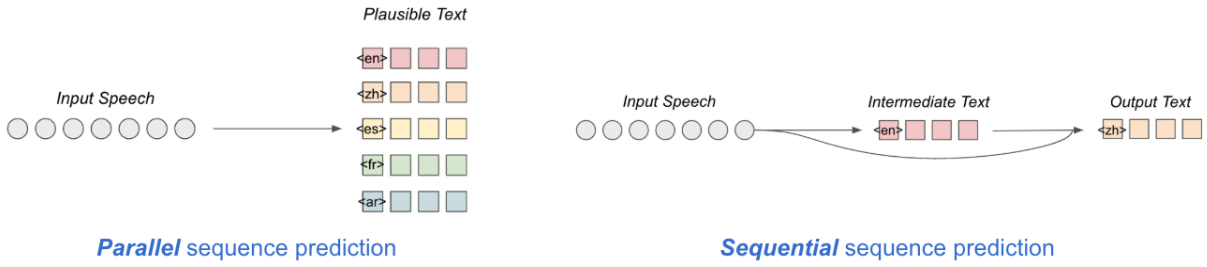

Figure 1.2: Parallel versus Sequential Multi-Sequence Modeling

In figure 1.2, we illustrate the computational forms of these two additional tasks of 1) determining what language is being spoken and 2) performing both transcription and translation. The former is a parallel process while the latter is a sequential process.

Key Idea

We described the probabilistic and computational forms of two additional tasks presented by multilingual speech processing: 1) determining what language is being spoken and 2) performing both transcription and translation - both are *multi*-sequence problems by nature.

Modeling either of these multilingual tasks with a direct *single* sequence prediction approach therefore has several risks, as follows:

- **Classification Error Propagation** - if a system incorrectly identifies the spoken language, then it is very likely to produce an incorrect transcription. This can be especially severe in cases where two phonetically related yet lexically distinct languages are confused.
- **Task Under-specification** - specifically in regards to the translation scenario, systems that do not perform an intermediate transcription step do not have any way to improve the ultimate translation by improving source language transcription, which is often a very useful lever.
- **Data Inefficiency** - training systems with overly monolithic architectures forces languages or tasks to compete for the same model capacity, which is a particularly limiting when building dedicated models for applications supported by limited amounts of supervised data.

Therefore, it warrants empirically assessing the severity of these issues across various system use cases - is the approximation, which saves computational cost, worth the loss in modeling accuracy? And how can we alternatively construct multi-sequence prediction models?

This thesis focuses on this line of questioning along two axes of multilinguality: a) *dedicated* use cases targeting local coverage of nuanced elements of human language, such as bilingual code-switching and dialectal diglossia, and b) *foundation* use cases targeting global coverage of many languages overall. The hope is that this research can lead to deeper and broader multilingual speech processing.

1.2 Taxonomy

This thesis is divided first into the parallel and sequential multi-sequence modeling types. Then within each type, we break applications down into two further settings - non-temporal and temporal. This results in four parts, which are as follows:

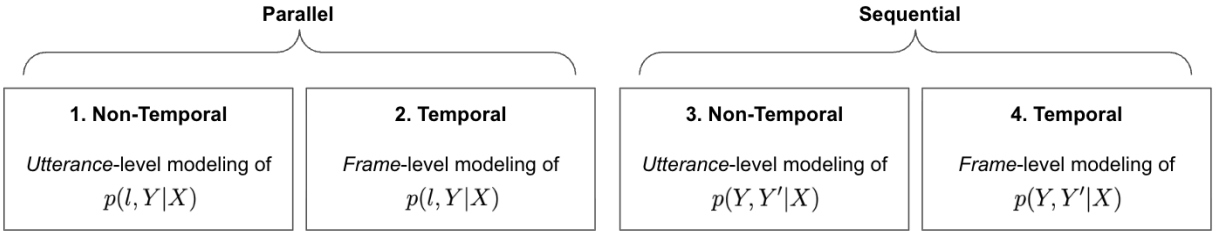

Figure 1.3: Thesis Taxonomy

- Parallel, Non-Temporal - refers to Multilingual Speech Recognition in the Wild, a setting that requires determining what language is being spoken at an utterance-level

- Parallel, Temporal - refers to Code-Switched Speech Recognition, a setting that requires determining what languages are being spoken and when they are being spoken. This includes both *inter-sentential* and *intra-sentential* code-switching. Inter-sentential code-switching occurs when language boundaries align with sentence or utterance boundaries (e.g., one sentence in English followed by another in Spanish), and can often be addressed by existing multilingual pipelines that combine language identification (LID) with monolingual ASR. In contrast, intra-sentential code-switching occurs within a single utterance, where languages are interleaved at the phrase or word level.

- Sequential, Non-Temporal - refers to Offline Speech Translation, a setting that requires performing transcription and translation of pre-recorded audio

- Sequential, Temporal - refers to Streaming Speech Translation, a setting that requires performing transcription and translation for a continuous speech in real-time with low latency

This thesis is organized following the taxonomy summarized in Figure 1.3. We cover dedicated model building and foundation model adaptation for each of the four aforementioned settings. Chapter in this thesis proposal document describe both published and proposed work, as well as reviews of prior work as necessary.

Chapter 2 (Parallel, Non-Temporal) covers Multilingual Speech Recognition in the Wild. We start by reviewing the first joint language identification and speech recognition study in the end-to-end neural network era, where a single sequence modeling paradigm was sufficient for 10 language speech recognition. We then shift to modern 100 language speech recognition foundation models, where transcription performance on tail languages is severely bottle-necked by error propagation from language misidentification. We propose to alleviate this issue via an inference-time adaptation, referred to as *Multilingual N-best Re-ranking*.

Chapters 3 through 7 (Parallel, Temporal) cover Code-Switched Speech Recognition. We start by evaluating architectures for building dedicated bilingual speech recognition models, such as for Singaporean Mandarin-English speakers. We then propose a different approach referred to as *Align-then-Stitch*. We then shift to evaluating foundation models on massively multilingual and code-switched speech, starting by constructing two new datasets, referred to as *CS-FLEURS*, which aims to understand whether speech systems trained on 100 languages can also process code-switched speech at that multilingual scale, and *CS-YODAS*, which aims to understand how code-switching appears in-the-wild in web-scale data. Finally, we scale the Align-then-Stitch approach towards supporting code-switching between 100 languages in the work referred to as the *CS-Anything* project.

Chapters 6 and 7 (Sequential, Non-Temporal) cover Offline Speech Translation. We start by evaluating architectures for building dedicated speech translation models, such as for translating

English speech into many languages and for Tunisian Arabic, a lower-resourced dialect. We then propose a different approach referred to as *Multi-Decoder*, an end-to-end differentiable cascade of speech recognition and machine translation sub-models.

Chapters 8 through 10 (Sequential, Temporal) cover Streaming Speech Translation. We start by evaluating architectures under the additional constraint of low-latency real-time prediction. We then propose several improvements, referred to as *Hierarchical CTC* and *Post-edit Operations*. We then shift to evaluating speech-enabled large language models, which are capable of chain-of-through prediction, under this same temporal constraint.

Part I

Parallel, Non-Temporal Multi-Sequence Modeling

Chapter 2

Multilingual Speech Recognition in the Wild

Summary

This chapter delves into **parallel, non-temporal** multi-sequence modeling, which is required for Multilingual Speech Recognition in the Wild.

Multilingual Automatic Speech Recognition (ASR) models are typically evaluated in a setting where the ground-truth language of the speech utterance is known, however, this is often not the case for most practical settings. Automatic Spoken Language Identification (SLID) models are not perfect and misclassifications have a substantial impact on the final ASR accuracy. In this chapter, we present a simple and effective N-best re-ranking approach which adapts models during inference-time for parallel multi-sequence prediction - this allows us to improve multilingual ASR accuracy for several prominent acoustic models by employing external features such as language models and text-based language identification models. Our results on FLEURS using the MMS and Whisper models show spoken language identification accuracy improvements of 8.7% and 6.1%, respectively and word error rates which are 3.3% and 2.0% lower on these benchmarks.[1]

2.1 Introduction

Recent work in multilingual automatic speech recognition (ASR) has drastically increased language coverage to support hundreds and even thousands of languages. This includes approaches based on labeled training data such as Whisper (Radford et al., 2023), USM (Zhang et al., 2023), Seamless (Communication et al., 2023) and MMS (Pratap et al., 2024) as well as zero-shot work (Li et al., 2022b; Zhao et al., 2024). The typical evaluation setting of these models assumes that the ground-truth language is known in advance to the model and we refer to this as the *lab* evaluation

[1] https://github.com/facebookresearch/fairseq/tree/main/examples/mms/lid_rerank

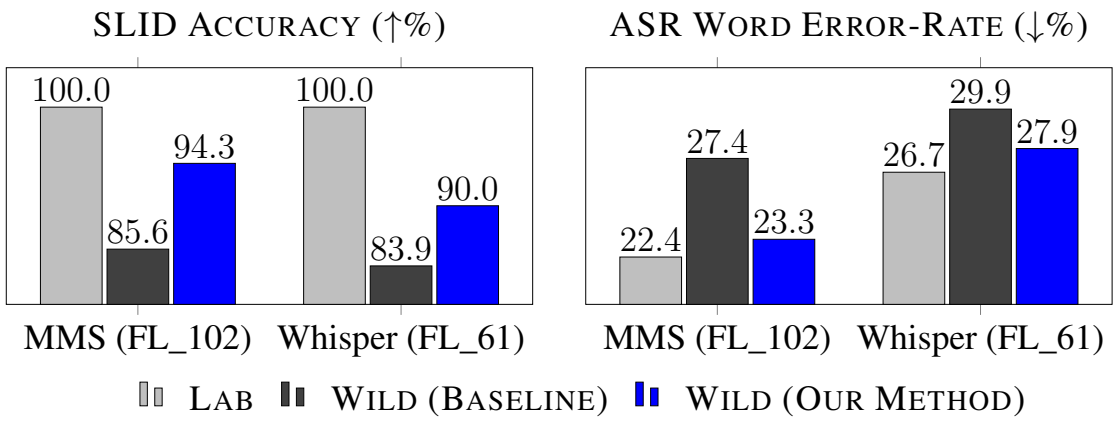

Figure 2.1: Multilingual ASR evaluation often assumes perfect spoken language identification (Lab) leading to much lower word error rates compared to real world spoken language identification (Wild - Baseline). Our method alleviates this lab-to-wild degradation (Wild - Our Method).

setting. However, in most practical settings, the language is not known and needs to be predicted using spoken language identification (SLID) - a setting we refer to as the *wild*.

SLID models (Radford et al., 2023; Pratap et al., 2024) perform well on high-resource languages such as English but less so on low-resource languages, e.g., for Whisper, there are 13 out of 100 supported languages which have a SLID accuracy of below 50% on FLEURS (Conneau et al., 2023). The errors of SLID bias dowstream ASR models towards the incorrect language and may result in unusable outputs. This typically results in a higher average word error rate for ASR but it is important to note that this increase is driven by high error rates on a subset of samples for which the output will be essentially unusable. Figure 2.1 illustrates the impact of imperfect SLID on ASR performance and how our method can recover much of that lost performance.

In this chapter, we propose a simple and effective N-best re-ranking approach to alleviate the error propagation from SLID to ASR (§2.2). The key idea is to defer the final SLID decision until *after* ASR has been performed for the top N SLID predictions. This enables the use of features operating over the resulting transcriptions to better determine the correct language. The set of features includes language models, written language identification models, as well as simple acoustic models to choose the final output (Figure 2.2 illustrates our method).

The results show that this method can substantially improve accuracy on both FLEURS (Conneau et al., 2023) and ML-SUPERB (Shi et al., 2023) for MMS and Whisper, as well as Seamless. Moreover, the method often approaches the ASR performance which could have been achieved with the correct SLID prediction. A downside of the method is the higher computational requirements, however, we find that even for small N-best list sizes of $N = 2$, the majority of the accuracy improvements can be realized.

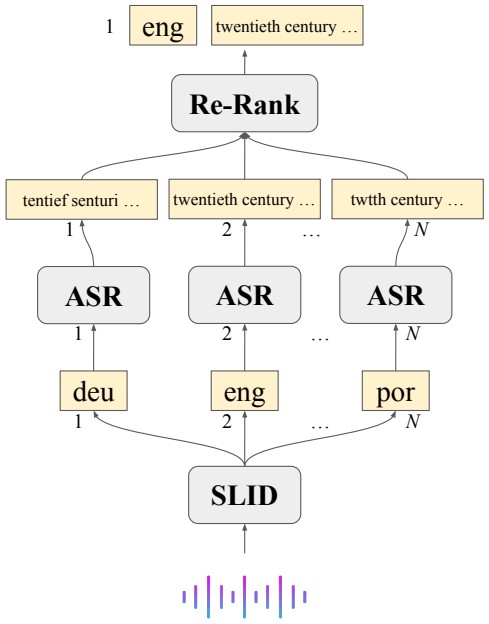

Figure 2.2: Illustration of our multilingual N-best re-ranking approach.

2.2 Proposed Method

We adapt the well-known N-best re-ranking framework typically used in monolingual ASR to multilingual ASR (§2.2). We then describe the set of quality estimation systems used to score and re-rank the multilingual N-best list (§2.2).

Multilingual ASR N-best Lists

N-best list re-ranking for a single language operates by using the ASR model to generate N-best candidate transcriptions and ranking them by their likelihood, for instance via beam search or sampling (Och, 2003; Chan et al., 2016; Salazar et al., 2020; Li et al., 2022a; Prabhavalkar et al., 2024). Finally, the candidates are then re-scored according to additional features and a new candidate is chosen. Crucially, the N candidate transcriptions are all in the same language.

The **multilingual N-best re-ranking** we consider operates differently: first, we perform spoken language identification to obtain the N highest scoring languages predicted by the SLID model for a given utterance. Next, we obtain the single highest scoring transcription for each of the N languages from the multilingual ASR system by conditioning the ASR model on each language. Finally, we place the resulting transcription for each language in the N-best list for re-ranking.

Features for Re-ranking

We utilize the following six **features for re-ranking**, each of which estimates a subset of the overall joint likelihood of a speech-language-transcript triplet, $P(S, L, T)$:

- $P(T)$: (1) Language Model (LM), (2) Text Length (Len)[2]
- $P(L, T)$: (3) Written LID Model (WLID)
- $P(S, L)$: (4) Spoken LID Model (SLID)
- $P(S, T)$: (5) ASR, (6) u-roman ASR (UASR)

where the SLID and ASR systems are the exact same systems which produced the N-best list, i.e., we do not use Whisper to re-rank MMS or vice-versa.

These features (aside from SLID) judge the quality of the transcripts T in different ways: the LM judges fluency of T in the respective language, the WLID model operates similarly but is more focused on individual words. Both can identify ASR outputs consisting of many non-word tokens such as when SLID is incorrect. u-roman ASR (Zhao et al., 2024) is an acoustic CTC model trained on romanized text (Hermjakob et al., 2018) which standardizes the text of any language to the standard Latin alphabet using a set of heuristics. The u-roman acoustic model penalizes candidates which are not faithful to the audio and we found this to be useful for ASR models known to hallucinate such as Whisper.

2.3 Experimental Setup

Spoken Lang ID and Automatic Speech Recognition Models

Table 2.1 shows the three multilingual ASR setups we chose for our study which are based on recent prominent open-sourced models. For Whisper (Radford et al., 2023), a single auto-regressive neural network first predicts SLID via a language token (e.g. [eng] or [spa]) which conditions the generation of subsequent ASR tokens. Joint prediction of SLID and ASR is a common approach in multilingual ASR (Watanabe et al., 2017a; Toshniwal et al., 2018; Hou et al., 2020b; Chen et al., 2023).

For MMS and Seamless, separate classification-based SLID models first predict SLID. Then MMS (Pratap et al., 2024) takes the language prediction by selecting language-specific CTC adapters, while Seamless (Communication et al., 2023) auto-regressively conditions ASR on a language token similar to Whisper.

[2]Not an actual likelihood estimator but can be used as a (weak) proxy.

Table 2.1: We test our method on three separate model setups, each consisting of Spoken Language Identification (SLID) and Automatic Speech Recognition (ASR) components.

NAME	LANGS	SLID COMPONENT	ASR COMPONENT
MMS (Pratap et al., 2024)	1162	MMS-1B	MMS-1B
Whisper (Radford et al., 2023)	100	Whisper v2 (joint SLID+ASR)	
Seamless (Communication et al., 2023)	96	ECAPA-TDNN (Ravanelli et al., 2021)	SeamlessM4T v2 (Communication et al., 2023)

We use greedy decoding for all ASR models. We do not use an external LM to decode the CTC-based MMS model.

Re-ranking Feature Models

We use 6 auxiliary re-ranking models to produce the 6 features for re-ranking introduced in §2.2:

- $P(T)$: (1) For LM we use MaLA-500 (Lin et al., 2024) which supports 534 languages and is based on Llama-2 7B. (2) For Len we use character count, including spaces.
- $P(L, T)$: (3) For WLID we use NLLB (Costa-jussà et al., 2022) which can classify text into 200 languages.
- $P(S, L)$: (4) For SLID we use the same model which produced the N-best languages in the particular setup.
- $P(S, T)$: (5) For ASR we use the same model which produced the N-best list in the particular setup. (6) Finally, for UASR we use MMS_Zero-Shot (Zhao et al., 2024) which supports force alignment with romanized text.

All of these models produce log likelihood scores with the exception of Text Length (Len).

Hyper-parameter Tuning

The re-ranking is determined by the combined feature score:

$$\text{score} = \sum_{i=1}^{F} w_i f_i \tag{2.1}$$

where f_i is the feature value of one of the $F = 6$ aforementioned models and w_i is the feature weight coefficient. The weights are tuned via random search over 10k iterations in which each

coefficient is randomly set within a defined range. This is achieved by computing the re-ranking WER on a development set for all 10k coefficient sets and then selecting the one which minimizes WER. We use the official dev sets of FLEURS and ML-SUPERB, tuning separately for each model and dataset. For LM, ASR, and UASR we use a range of 0 to 10. For SLID and WLID we use a range of 0 to 100. For Text Length we use a range of -5 to 5.

Datasets and Evaluation

We evaluate on FLEURS (FL; (Conneau et al., 2023)) and ML-SUPERB (MS; (Shi et al., 2023)) using the languages supported by each model. For instance, MMS is evaluated on FL_102, the full benchmark of 102 languages, while Whisper is evaluated on FL_61, the supported subset. Therefore, MMS results are not directly comparable to Whisper. We made this choice to evaluate our approach for each model as broadly as possible.

We report both SLID and ASR performance, where the former is measured in terms of accuracy and the latter is measured in terms of word error rate (WER) for all languages except character-based languages for which character error rate is used.[3] We compare our method to the baseline 1-best SLID+ASR performance as well as the topline (oracle) re-ranking performance. The oracle is based on selecting the candidate with the correct language if it exists in the N-best list. If the correct language is not in the N-best list, then the candidate with the highest SLID score is selected.

2.4 Results and Analysis

We first report the accuracy of N-best re-ranking on Whisper, MMS, and Seamless both for SLID and ASR (§2.4), analyze potential regressions (§2.4), and improvements on tail langauges (§2.4). Next, we ablate the relative importance of each re-ranking feature (§2.4). Finally, we show that our method is still effective with small values of N (§2.4) and outperforms standard monolingual N-best re-ranking (§2.4).

Main Results

Table 2.2 shows that the method **increases SLID accuracy on average by 5% absolute** on FLEURS and ML-SUPERB and three model setups. The approach captures about 41% of the oracle performance.

Table 2.3 shows the corresponding ASR performance. 10-best re-ranking **reduces the WER by nearly 3% absolute**, or about 9% relative, and captures about 79% of the oracle which has access to the real language ID.

[3]Character langs: adx, bod, cmn, dzo, jpn, khg, khm, lao, mya, tha, yue.

Table 2.2: SLID Accuracy (↑%): Re-rank 10-best (our method) vs. baseline (1-best) and oracle (10-best).

	MMS		WHISPER		SEAMLESS		Avg
	FL_102	MS_143	FL_61	MS_76	FL_73	MS_88	
Baseline	85.6	80.3	83.9	76.7	81.4	69.6	79.6
Re-rank	94.3	85.0	90.0	79.5	83.1	75.8	84.6
Oracle	96.7	93.0	98.6	92.7	86.8	83.6	91.9

Table 2.3: ASR Word Error Rate (↓%): Re-rank 10-best (our method) vs. baseline (1-best) and oracle (10-best).

	MMS		WHISPER		SEAMLESS		Avg
System	FL_102	MS_143	FL_61	MS_76	FL_73	MS_88	
Baseline	26.6	29.7	29.9	33.8	28.5	32.8	30.2
Re-rank	23.3	26.9	27.9	30.9	26.2	28.9	27.4
Oracle	23.0	26.4	27.7	30.6	25.2	26.6	26.6

Generally, performance is better on FLEURS than ML-SUPERB which we attribute to the latter being a harder benchmark. The greater improvements in ASR than in SLID are due only in part to our choice to minimize ASR WER during coefficient tuning. Another factor is that in some cases a model can produce better outputs (lower WER) for an incorrect language than for the correct one, but this is rare (more often on languages with high WER in the lab setting).

Drilldown into Main Results

The results so far showed an overall improvement but is there any regression on the samples which were originally correctly classified by SLID?

To better understand this issue, we divide SLID and ASR performance into samples classified correctly and incorrectly by the original SLID. Table 2.4 shows that the correct subset has slight SLID accuracy regression of 1% and a 0.1% WER degradation for ASR, while the SLID accuracy of the error subset increased from 0% to nearly 67% with a WER reduction of nearly 29% absolute. It is important to note that the correct subset is much larger (85.6% vs. 14.4%) and regressions there have a larger impact. Nonetheless, the regression is still very small and is accompanied by substantial improvements on the error subset. Moreover, the oracle shows that the error subset is

Table 2.4: Performance Breakdown into originally correct SLID vs. originally incorrect SLID for MMS FL_102. There is only a very small regression on the correct subset and large improvements on the incorret subset.

	ORIG. CORRECT SLID (85.6% OF TOTAL)		ORIG. INCORRECT SLID (14.4% OF TOTAL)	
System	Acc (↑%)	WER (↓%)	Acc (↑%)	WER (↓%)
Baseline	100.0	20.8	0.0	69.1
Re-rank	99.0	20.9	66.7	40.4
Oracle	100.0	20.8	76.9	39.3

hard: the oracle accuracy of 39.3% WER is nearly double that of the correct subset (20.8%).

Effect on Tail Languages

The baseline SLID accuracy is not uniform across languages - does the method alleviate the problems at the tail?

Table 2.5 shows the SLID and ASR performance for the ten FLEURS languages with the lowest baseline SLID accuracy for MMS as well as the improvements with our method. Our method increases the average SLID accuracy from 18.1% to 83.1% and decreases average WER from 67.4% to 39.3%. See for instance the Odia language - SLID accuracy increases by 75% absolute while WER decreases by 48% absolute.

Feature Ablation

To better understand the relative importance of each of the six features we use for re-ranking, we conduct the following experiment: starting from an empty set we add the feature which results in the best performance after tuning coefficients to minimize WER, repeating until all features have been added. The rankings for MMS FL_102 and Whisper FL_61 are reported in Table 2.6.

The most important feature for both MMS and Whisper is SLID which is in part because it is the model which originally produces the list of the N-best languages. The remaining feature rankings show some variation across models which we believe is due to the different nature of the models and their different error patterns. For instance, the uroman acoustic model (UASR) is the second most useful feature for Whisper but the least useful for MMS. This is likely because Whisper tends to generate unrelated transcriptions and UASR is effective in determining this since

Table 2.5: Tail Performance: Re-ranking impact on the ten lowest performing languages (by SLID Acc) for MMS FL_102.

	SLID Acc (↑%)		ASR WER (↓%)	
Language	Baseline	Re-rank	Baseline	Re-rank
Xhosa	0.0	76.1	56.4	36.2
Kabuverdianu	1.2	40.4	52.4	34.4
Irish	5.3	96.0	96.9	62.4
Occitan	13.6	96.2	48.8	34.5
Odia	18.4	93.7	87.1	39.6
Kyrgyz	23.6	93.2	32.8	19.0
Central Kurdish	28.0	86.1	74.3	44.8
Northern Sotho	29.6	99.4	75.3	27.3
Igbo	30.3	63.5	72.7	52.0
Umbundu	30.7	86.4	77.3	42.8
Average	18.1	83.1	67.4	39.3

UASR is a CTC model which is force-aligned with the Whisper transcription. On the other hand, MMS is already a CTC model and this capability is therefore redundant.

Written LID (WLID) is the second most important feature for MMS because the CTC model is more likely to output non-words given the incorrect language tag and WLID detects this effectively. The acoustic model score (ASR) is useful for both models since it is measure of confidence in the transcriptions. Finally, an external language model is most redundant for Whisper, since it has a strong built-in language model already.

N-best List Size Ablation

A drawback of the method is an increase in computational cost proportional to the size of the N-best list. To better understand the impact of N, we measure performance for different values (where $N = 1$ corresponds to the baseline).

Figure 2.3 shows that the larger values of N result in better accuracy but the largest marginal improvement is achieved with just $N = 2$ which captures 46% and 61% of the SLID and ASR improvements of $N = 10$ for MMS and 55% and 75% for Whisper, respectively. This suggests that a lightweight version of the proposed method may be preferable if computational cost is a limiting factor.

Table 2.6: Feature Importance Rankings: MMS FL_102 and Whisper FL_61.

MMS (FL_102)			WHISPER (FL_61)		
Rank	Signal	WER (↓)	Rank	Signal	WER (↓)
1	SLID	27.0	1	SLID	29.9
2	+WLID	25.1	2	+UASR	29.2
3	+ASR	24.5	3	+ASR	28.8
4	+LM	24.1	4	+WLID	28.0
5	+Len	23.7	5	+Len	28.0
6	+UASR	23.7	6	+LM	27.9

Table 2.7: Multilingual vs. Monolingual N-best Lists: Whisper FL_61

System	Acc (↑%)	WER (↓%)
Baseline (1-best)	83.9	29.9
Monolingual Re-rank (10-best)	83.9	28.7
Multilingual Re-rank (10-best)	90.0	27.9

Comparison to Standard Monolingual N-best Lists

Finally, we compare multilingual re-ranking to standard monolingual re-ranking which is based on an N-best list obtained via beam search for the most likely LID prediction of an utterance. For this comparison, we use the same set of re-ranking features for each type of N-best list.

Table 2.7 shows that monolingual re-ranking outperforms the 1-best baseline but our approach further improves ASR by 0.8% absolute WER while also improving SLID accuracy by 6.1% absolute. This shows that a large part of the improvement is due to identifying the correct language and not just due to switching to an n-best re-ranking setup. Moreover, our method may also benefit from considering multiple ASR candidates per language and future work may investigate this.

2.5 Related Work

The idea of determining language ID after examining ASR outputs is not new. Prior work have employed this to improve the LAS architecture (Toshniwal et al., 2018; Li et al., 2018). Other works focused on code-switched ASR in particular have also explored various forms of SLID and ASR conditioning (Yan et al., 2022b, 2023c; Hussein et al., 2024a). (Metze et al., 2000) uses

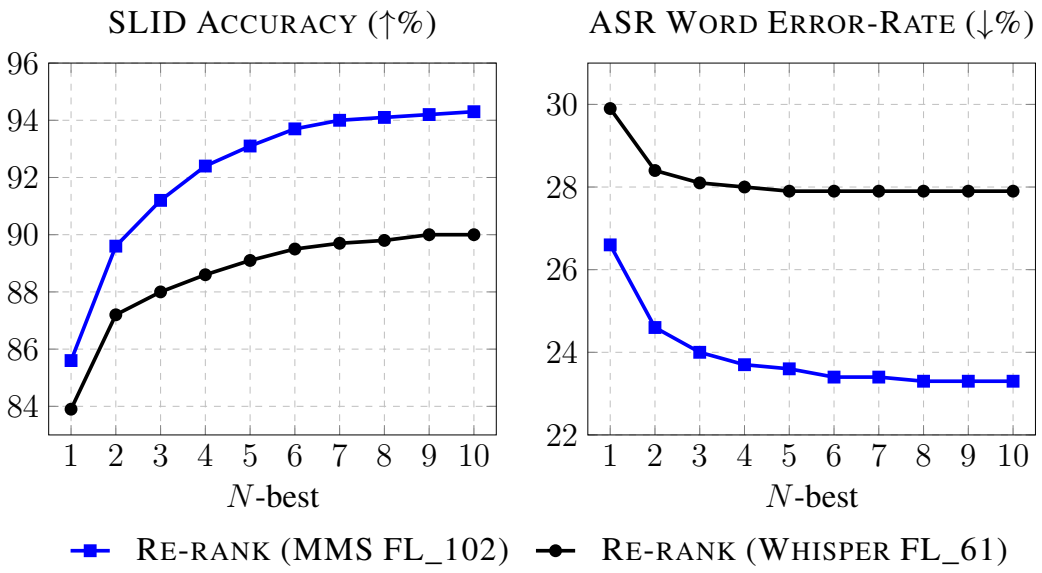

Figure 2.3: Effect of different sized N-best lists for SLID (left) and ASR (right).

ASR confidence to distinguish between only three possible languages, an early precursor to our work. (Wang et al., 2019) uses a set of features like we do but they propose either lattice regression or neural network based methods for combining them. Finally, (Chandak et al., 2020) eschews the use of any features and instead learn latent representations of ASR candidates via a bespoke architecture.

2.6 Conclusion

This chapter presents a simple N-best re-ranking approach to improve multilingual ASR using a small set of external features. Empirical results show spoken language identification accuracy improvements with the Whisper and MMS models of 8.7% and 6.1%, respectively and word error rate reductions of 3.3% and 2.0% on the FLEURS benchmark. Moreover, we show that our method substantially benefits many languages which previously had very low SLID and ASR performance.

We've now established the efficacy of parallel multi-sequence modeling for this utterance-level (or non-temporal) language identification dependent ASR task. Next, let's consider the case that language identity is variable within an utterance as in code-switched speech.

Part II

Parallel, Temporal Multi-Sequence Modeling

Chapter 3

Align-then-Stitch: Joint Code-Switched and Monolingual Speech Recognition

Summary

This chapter delves into **parallel, temporal** multi-sequence modeling which is required for code-switched speech recognition. This chapter introduces the general problem setting: bilingual speech recognition, where systems must process monolingual speech and intra-sententially code-switched speech indiscriminately.

Conversational bilingual speech encompasses three types of utterances: two purely monolingual types and one intra-sententially code-switched type. In this chapter, we propose a general *Align-then-Stitch* framework to jointly model the likelihoods of the monolingual and code-switch sub-tasks that comprise bilingual speech recognition. By defining the monolingual sub-tasks with label-to-frame synchronization (the *Align* stage), our joint modeling framework can be *conditionally factorized* such that the final bilingual output, which may or may not be code-switched, is obtained given only monolingual information (the *Stitch* stage). We show that this conditionally factorized joint framework can be modeled by an end-to-end differentiable neural network. We demonstrate the efficacy of our proposed model on bilingual Mandarin-English speech recognition across both monolingual and code-switched corpora.

3.1 Introduction

Conversational spoken language defies monolithic form, but rather is highly adaptive to situational cues such as who you are speaking to or what you are speaking about (Auer, 2013; Dragojevic et al., 2015). For instance, bilingual speakers often code-switch between different languages to

facilitate communication (Heredia and Altarriba, 2001; Hou et al., 2020a). In fact, the act of bilingual code-switching itself may convey various aspects about the speaker such as a desire to spark an interpersonal connection (Ahn et al., 2020), a level of subject-matter expertise (Yoder et al., 2017), or an affinity to some socio-economic status (Liu, 2019).

In order to broadly cover bilingual speech, recognition systems need to recognize not only monolingual utterances from two different languages, but also intra-sententially code-switched (CS) utterances where both languages are present (Heredia and Altarriba, 2001; Hou et al., 2020a). While recent advancements in the related field of multilingual speech recognition have significantly improved the language coverage of a single system by training on mixtures of monolingual utterances (Adams et al., 2019; Pratap et al., 2020a), these works typically do not account for intra-sentential CS. Prior works have adapted large-scale multilingual models to more flexibly identify language switch points (Li et al., 2019a; Zhou et al., 2022a), but performance is dependent on the cross-lingual dynamics of the selected languages (Seki et al., 2018).

Another approach is to directly optimize towards intra-sentential CS for a particular bilingual pair. A significant portion of recent works are ameliorating the linguistic differences between two unrelated languages by explicitly defining cross-lingual phone-merging rules (Lyudovyk and Pylypenko, 2014; Luo et al., 2018; Sivasankaran et al., 2018) or by implicitly learning latent language identity representations (Yılmaz et al., 2016; Song et al., 2017; Zeng et al., 2018; Kim and Seltzer, 2018; Shan et al., 2019b; Zhang et al., 2021b). The other significant portion of recent works are ameliorating the scarcity of paired CS speech data through data efficient methods that incorporate monolingual data into both acoustic (Lu et al., 2020; Zhou et al., 2020b; Dalmia et al., 2021a; Zhang et al., 2021a) and language modeling (Gonen and Goldberg, 2018; Shan et al., 2019a; Chuang et al., 2020; Dalmia et al., 2021a; Zhang et al., 2021a) as well as through data augmentation techniques that generate synthetic CS data (Chang et al., 2018; Pratapa et al., 2018; Ma et al., 2019b; Lee et al., 2019). However, works focusing narrowly on intra-sentential CS often ignore or sacrifice performance on monolingual scenarios (Shah et al., 2020; Sitaram et al., 2019)

We are interested conversational bilingual speech recognition systems (ASR) that can cover both monolingual and intra-sententially CS scenarios. In particular, we are interested in systems that can 1) indiscriminately recognize both monolingual and intra-sententially CS utterances, 2) efficiently leverage monolingual and CS ASR training data, and 3) be built in an end-to-end manner.

We first propose a formulation of the bilingual ASR problem as a *conditionally factorized* joint model of monolingual and CS ASR where the final output is obtained given only monolingual label-to-frame synchronized information §3.3. We then apply an end-to-end differentiable neural network, which we call the Conditional RNN-Transducer (RNN-T), to model our conditional joint formulation §3.3. We show the efficacy of the Conditional RNN-T model in both monolingual and CS scenarios compared to several baselines §3.5. Next, we demonstrate the Language-Separation

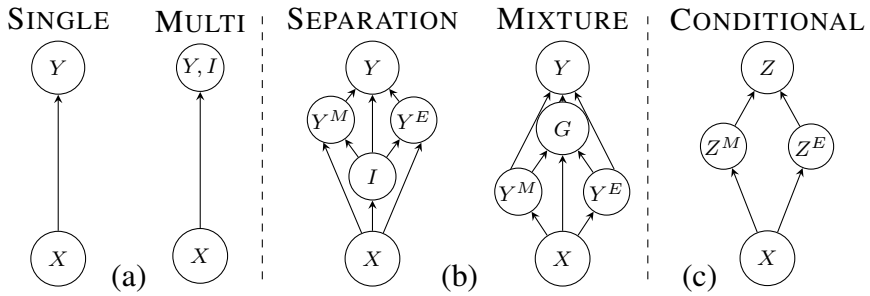

Figure 3.1: Probabilistic graphical model representations of different formulations of the bilingual ASR task, separated into three categories: (a) direct, (b) divide-and-conquer, and (c) conditional.

ability, an added benefit of our proposed model §3.5. Finally, we validate a key conditional independence assumption in our framework by showing experimentally that given monolingual label-to-frame information, no other information from the observation is required §3.5.

3.2 Background and Motivation

In this section, we interpret the underlying probabilistic graphical models of prior works in bilingual ASR, as applied to Mandarin-English, to motivate our conditionally factorized framework in §3.3.

Direct Graphical Models

Bilingual ASR is a sequence mapping from a T-length speech feature sequence, $X = \{\mathbf{x}_t \in \mathbb{R}^D | t = 1, ..., T\}$, to an N-length label sequence, $Y = \{y_n \in (\mathcal{V}^M \cup \mathcal{V}^E) | n = 1, ..., N\}$ consisting of Mandarin $\mathcal{V}^M$ and English $\mathcal{V}^E$. As shown in Figure 3.1.a, the simplest approach models the bilingual output Y as a random variable with a *single* dependency on the observation X. Both the Mandarin $\mathcal{V}^M$ and English $\mathcal{V}^E$ vocabularies are regarded as part of the same set of output units and no explicit distinctions are made between the languages. This unification of two languages from unrelated families complicates the ASR task via phonetic ambiguities, but hand-crafting phone-merging rules may alleviate this issue (Lyudovyk and Pylypenko, 2014; Luo et al., 2018; Sivasankaran et al., 2018).

Alternatively, *multi*-tasking with language-identification can induce useful latent distinctions between Mandarin and English representations. Here, we introduce language ID I as another random variable with a single dependency on the observation X. Note that the bilingual output Y and

the language-ID I do not interact with each other aside from being implicitly related as they both directly depend on X. During training, this multi-tasking approach helps to resolve cross-lingual ambiguities within the shared bilingual latent representations (Yılmaz et al., 2016; Song et al., 2017; Zeng et al., 2018; Kim and Seltzer, 2018; Shan et al., 2019b; Zhang et al., 2021b). During inference, the bilingual output is obtained directly from the observation similar to the single dependency approach. The main drawback of these direct approaches is the potentially high complexity of the $X \rightarrow Y$ dependency due to conflicting linguistics of different two languages.

Divide-and-Conquer Graphical Models

Alternatively, the bilingual ASR task can be decomposed into its sub-component monolingual parts as in the divide-and-conquer approaches shown in Figure 3.1.b. The *separational* approach uses language ID I to segment the observation X into monolingual parts before passing those to monolingual recognizers which separately predict Mandarin $Y^M = \{y_l^M \in \mathcal{V}^M | l = 1, ..., L\}$ and English $Y^E = \{y_l^E \in \mathcal{V}^E | l = 1, ..., L\}$. Finally, the monolingual outputs Y^M and Y^E are stitched together accordingly to obtain the final bilingual output Y. This approach successfully introduces simpler $X \rightarrow Y^M$ and $X \rightarrow Y^E$ dependencies, but does so at the cost of the additional dependencies associated with the language ID random variable I: $I \rightarrow Y^M$, $I \rightarrow Y^E$, and $I \rightarrow Y$. Therefore, performance becomes highly dependent on the ability to correctly predict the language identity at a segment level (Chan et al., 2004; Weiner et al., 2012; Lyu et al., 2013; Rallabandi et al., 2018; Li et al., 2019b).

The *mixture*-based approach avoids this dependency on the language ID I by instead introducing another random variable G, a gating mechanism. This approach first noisily models $X \rightarrow Y^M$ and $X \rightarrow Y^E$, but maintains latent representations of each so as to avoid early decisions. Then, it models the gating mechanism G as a function of these latent monolingual representations; we approximate this dependency in our graph as $(X, Y^M, Y^E) \rightarrow G$. Finally, the gating mechanism G is used to fuse the latent monolingual representations to ultimately obtain the bilingual output Y. This is a promising approach that efficiently combines information from monolingual experts by operating entirely within the latent space (Lu et al., 2020; Zhou et al., 2020b; Dalmia et al., 2021a). However, mixture-based approaches much like their separational cousins incur the cost of the additional dependencies associated with the gating random variable G. Our motivation is to decompose the bilingual ASR problem into monolingual sub-tasks without incurring any additional random variables.

Table 3.1: Results comparing the Conditional RNN-T models to Vanilla and Gating RNN-T baselines on intra-sententially code-switched (CS), monolingual Mandarin, and monolingual English test sets. The upper half shows results when using only CS data during fine-tuning while the bottom half shows results when using CS + monolingual data during fine-tuning. All models are pre-trained on monolingual data.

Model Type	Model Name	Pre-trained Encoder(s)	Fine-tuning Data	CODE-SWITCHED MER	CODE-SWITCHED CER	CODE-SWITCHED WER	MONO-MAN CER	MONO-ENG WER
Direct	Vanilla RNN-T	✓	CS	12.3	9.9	34.3	17.9	81.4
Mixture	Gating RNN-T	✓	CS	11.5	9.1	33.0	17.7	**78.3**
Conditional	Our Proposed Model	✓	CS	11.5	9.1	33.2	15.5	82.9
Conditional	+ Language-Separation (LS)	✓	CS	**11.1**	**8.7**	**32.7**	**15.3**	82.7
Direct	Single RNN-T	✓	CS + M	11.3	9.3	30.8	6.5	17.8
Mixture	Gating RNN-T	✓	CS + M	11.2	8.8	34.7	5.7	34.6
Conditional	Our Proposed Model	✓	CS + M	10.3	8.2	29.5	5.4	16.5
Conditional	+ Language-Separation (LS)	✓	CS + M	**10.2**	**8.1**	**29.2**	**5.3**	**16.3**

3.3 Proposed Framework

In this section, we first propose a framework using label-to-frame synchronization to obtain bilingual outputs given only conditional monolingual information. Then we show an end-to-end differentiable method to model our conditionally factorized framework.

Conditionally Factorized Formulation of Bilingual ASR

Rather than treating CS ASR as a single sequence transduction task, we seek to decompose it into three portions: 1) recognizing Mandarin 2) recognizing English and 3) composing recognized monolingual segments into a bilingual sequence which may or may not be CS. However, given only monolingual portions of the output Y^M and Y^E we cannot form Y without the order in which they should be composed. Therefore, in order to satisfy our desired *conditional* probabilistic graph (shown in Figure 3.1.c), we need richer monolingual representations which contain ordering information.

Consider that for each T-length observation sequence X and L-length bilingual label sequence Y, there are a number of possible T-length label-to-frame sequences $Z = \{z_t \in \mathcal{V}^M \cup \mathcal{V}^E \cup \{\varnothing\} | t = 1 \dots T\}$. Here, z_t is either a surface-level unit or a blank symbol denoting a null emission as in Connectionist Temporal Classification (CTC) (Graves et al., 2006). The posterior of the

bilingual label sequence $p(Y|X)$ is then factorized as follows:

$$p(Y|X) = \sum_{Z \in \mathcal{Z}(Y)} p(Z|X) \tag{3.1}$$

Note that the summation is over all possible label-to-frame alignments $Z \in \mathcal{Z}(Y)$ for a given observation X and label Y pair.[1]

Next, we re-formulate any bilingual label-to-frame sequence in terms of its constituent monolingual label-to-frame sequences $Z^M = \{z_t^M \in \mathcal{V}^M \cup \{\varnothing\} | t = 1 \dots T\}$ and $Z^E = \{z_t^E \in \mathcal{V}^E \cup \{\varnothing\} | t = 1 \dots T\}$. Now, we can indeed obtain the bilingual Z given only conditional monolingual information Z^M and Z^E:

$$z_t = \begin{cases} z_t^M, & \text{if } z_t^M \in \mathcal{V}^M \text{ and } z_t^E = \varnothing \\ z_t^E, & \text{if } z_t^E \in \mathcal{V}^E \text{ and } z_t^M = \varnothing \\ \varnothing, & \text{if } z_t^M = \varnothing \text{ and } z_t^E = \varnothing \end{cases} \tag{3.2}$$

Note that at position t only one of z_t^M or z_t^E may be a surface-level unit while the other must be the blank symbol by definition. It is possible that both z_t^M and z_t^E are blank symbols, but it is not possible that both are surface-level units.

Following this interpretation of the bilingual sequence Z in terms of its monolingual parts Z^M and Z^E, we can re-formulate the posterior $p(Z|X)$ as a joint likelihood $p(Z, Z^M, Z^E|X)$:

$$\begin{aligned} p(Z|X) &= p(Z, Z^M, Z^E|X) && (3.3) \\ &= p(Z|Z^M, Z^E, X) p(Z^M, Z^E|X) && (3.4) \\ &\approx p(Z|Z^M, Z^E, \cancel{X}) p(Z^M|X) p(Z^E|X) && (3.5) \end{aligned}$$

From Eq. equation 3.4 to Eq. equation 3.5, we make two key independence assumptions. The first is that given Z^M and Z^E, no other information from the observation X is required to determine Z since the monolingual label-to-frame sequences already contain ordering information. We show that this assumption holds experimentally in §3.5. The second assumption is that $p(Z^M|X)$ and $p(Z^E|X)$ are independent, allowing for separate modeling of monolingual posteriors.

[1] Z maps to Y deterministically via repeat and blank removal rules (Graves, 2012a).

Conditional RNN Transducer

Monolingual Modules

The monolingual label-to-frame posteriors $p(Z^M|X)$ and $p(Z^E|X)$ are further factorized with the chain-rule as follows:

$$p(Z^M|X) \approx \prod_{t=1}^{T} p(z_t^M|X, \cancel{z_{1:t-1}^M}) \tag{3.6}$$

$$p(Z^E|X) \approx \prod_{t=1}^{T} p(z_t^E|X, \cancel{z_{1:t-1}^E}) \tag{3.7}$$

Note that we make the conditional independence assumption here as a modeling choice, mitigating label / exposure bias issues (Ranzato et al., 2016; Bottou et al., 1997).

Our proposed Conditional RNN-T models the posterior as follows. Given the observed speech feature sequence X, a Mandarin-only ENCODER_M maps to a sequence of hidden representations $\mathbf{h}^M = \{\mathbf{h}_t^M \in \mathbb{R}^D | t = 1, ..., T\}$ and an English-only ENCODER_E maps to a sequence of hidden representations $\mathbf{h}^E = \{\mathbf{h}_t^E \in \mathbb{R}^D | t = 1, ..., T\}$. Separate linear projection layers followed by softmax activations, SOFTMAXOUT_M and SOFTMAXOUT_M, yield the posteriors $p(z_t^M|X)$ and $p(z_t^E|X)$. We train these modules using CTC loss functions $\mathcal{L}_{\text{M\_CTC}}$ and $\mathcal{L}_{\text{E\_CTC}}$ (Graves et al., 2006).

Bilingual Module

The bilingual conditional likelihood $p(Z|Z^M, Z^E)$ is further factorized with the chain-rule as well:

$$p(Z|Z^M, Z^E) = \prod_{i=1}^{T+L} p(z_i|Z^M, Z^E, z_{1:i-1}) \tag{3.8}$$

The monolingual alignment information is passed to the bilingual module via the hidden representations $\mathbf{h}^M$ and $\mathbf{h}^E$. We therefore approximate $P(z_i|Z^M, Z^E, z_{1:i-1})$ as follows:

$$\mathbf{h}_t^{\text{ENC}} = \mathbf{h}_t^M + \mathbf{h}_t^E \tag{3.9}$$

$$\mathbf{h}_l^{\text{DEC}} = \text{DECODER}(z_{1:l-1}) \tag{3.10}$$

$$\mathbf{h}_{t,l}^{\text{JNT}} = \text{JOINT}(\mathbf{h}_t^{\text{ENC}}, \mathbf{h}_l^{\text{DEC}}) \tag{3.11}$$

$$p(z_i|\mathbf{h}^M, \mathbf{h}^E, z_{1:i-1}) = \text{SOFTMAXOUT}(\mathbf{h}_{t,l}^{\text{JNT}}) \tag{3.12}$$

Note that equation 3.12 approximates $p(z_i|Z^M, Z^E, z_{1:i-1})$ from equation 3.8 by using the latent

representations $\mathbf{h}^M$ and $\mathbf{h}^E$ to pass the monolingual label-to-frame information Z^M and Z^E.[2] We train these modules using the RNN-T loss function $\mathcal{L}_{\text{RNNT}}$ (Graves, 2012a).

Full Network

With Equations equation 3.1 and equation 3.5, the posterior $p(Y|X)$ finally becomes:

$$p(Y|X) \approx \underbrace{\sum_{\mathcal{Z}} p(Z|Z^M, Z^E)}_{\triangleq p_{\text{rnnt}}(Y|Z^M,Z^E)} \underbrace{\sum_{\mathcal{Z}^M} p(Z^M|X)}_{\triangleq p_{\text{ctc}}(Y^M|X)} \underbrace{\sum_{\mathcal{Z}^E} p(Z^E|X)}_{\triangleq p_{\text{ctc}}(Y^E|X)} \tag{3.13}$$

where the monolingual CTC, $p_{\text{ctc}}(Y^M|X)$ and $p_{\text{ctc}}(Y^E|X)$, and bilingual RNN-T, $p_{\text{rnnt}}(Y|Z^M, Z^E)$, objective functions are defined as summations over all possible frame-to-label sequences $Z^M \in \mathcal{Z}^M(Y^M)$, $Z^E \in \mathcal{Z}^E(Y^E)$, and $Z \in \mathcal{Z}(Y)$ respectively.

We train our Conditional RNN-T model using an initial monolingual pre-training step to maximize $p_{\text{ctc}}(Y^M|X)$ and $p_{\text{ctc}}(Y^E|X)$. We then fine-tune the entire network to maximize $p_{\text{rnnt}}(Y|Z^M, Z^E)$, including both the bilingual and pre-trained monolingual modules, on a mixture of monolingual and CS data. Since this first regime only implicitly applies monolingual conditioning via pre-trained initialization, we propose an explicit alternative regime which we call the Conditional RNN-T + Language-Separation (LS). This variant of our proposed model uses a multi-tasked loss $\mathcal{L}_{LS}$ with tunable λ:

$$\mathcal{L}_{\text{LS}} = \lambda\mathcal{L}_{\text{RNNT}} + (1-\lambda)(\mathcal{L}_{\text{M\_CTC}} + \mathcal{L}_{\text{E\_CTC}}) \tag{3.14}$$

The monolingual ground truths Y^M and Y^E are obtained by applying a language-specific mask to the bilingual ground truth Y.[3]

3.4 Data and Experimental Setup

Data: We use 200h of intra-sententially CS training data from the ASRU 2019 shared task where Mandarin is the matrix and English is the embedded language (Shi et al., 2020). We use 500h of monolingual Mandarin data for pre-training and a 200h subset for fine-tuning (Shi et al., 2020). We use 700h of accented monolingual English data for pre-training and a 200h subset for fine-tuning (Speechocean, 2017). We use provided test CS data and generate our own splits for monolingual test sets.

[2]Alternatively, logit or softmax normalized could be used here (Dalmia et al., 2019).

[3]This masking is applicable to both monolingual and CS Y. E.g. if $y_l \in \mathcal{V}^M \forall l$, then the masked Y^E is the empty string and Y^M is the entire Y.

Experimental Setup: All our models were trained using the ESPnet toolkit (Watanabe et al., 2018). Our input features are global mean-variance normalized 83 log-mel filterbank and pitch features (Povey et al., 2011). We apply the Switchboard Strong (SS) augmentation policy of SpecAugment (Park et al., 2019). We combine 5000 Mandarin characters with 5000 English BPE (Sennrich et al., 2016) units to form the output vocabulary. ENCODERS are conformers (Gulati et al., 2020; Guo et al., 2021) with 12 blocks, kernel size of 15, 2048 feed-forward dim, 256 attention dim, and 4 heads. DECODERS are LSTMs (Hochreiter and Schmidhuber, 1997; Watanabe et al., 2018) with 1 layer, 1024 embed dim, 512 unit dim, and 512 joint dim. The direct RNN-T baseline with only 1 encoder uses a doubled 512 attention dim, so all models have about 100M params. We use the Adam optimizer to train 80 epochs with an inverse square root decay schedule, a transformer-lr scale (Watanabe et al., 2018) of 1, 25k warmup steps, and an effective batchsize of 192. ENCODERS are pretrained using the hybrid CTC/Attention framework (Watanabe et al., 2017b). We use beam-size of 10 during inference. Ablation studies on CTC sub-nets use greedy decoding.

3.5 Results

In Table 3.1, we compare the CS and monolingual performances of our Conditional RNN-T and Conditional RNN-T + LS models to direct and mixture-based baselines, which are our re-implementations of Vanilla RNN-T (Dalmia et al., 2021a; Zhang et al., 2021b) and Gating RNN-T (Lu et al., 2020; Dalmia et al., 2021a) described in prior works. The top and bottom portions of Table 3.1 compare results when using only CS data versus using both CS and monolingual data during fine-tuning. Not only did all models perform significantly better on monolingual sets when using monolingual fine-tuning data, they also improved on the CS set suggesting that the monolingual data is indeed supplementing the CS training data.[4]

As shown in the bottom half of Table 3.1, the Gating RNN-T slightly outperforms the Vanilla RNN-T on CS and monolingual Mandarin but is degraded on monolingual English, suggesting that the gating mechanism overly focuses on outputting Mandarin due to the high skew of the CS training data towards Mandarin. Our proposed Conditional RNN-T model outperforms the best baselines consistently across evaluation sets. On the monolingual sets, the Conditional RNN-T model performs similarly to monolingual-only models trained on the same data. Further, the Language-Separation loss incrementally improves across all sets suggesting a benefit to the explicit monolingual conditioning method described in Eq. equation 3.14. We examine this benefit further in the subsequent sub-section.

[4]Unlike the shared task in (Shi et al., 2020), we evaluate on monolingual corpora and do not use the provided 3-gram LM, the full monolingual data during fine-tuning, data augmentation besides SpecAugment, or LID multi-tasking.

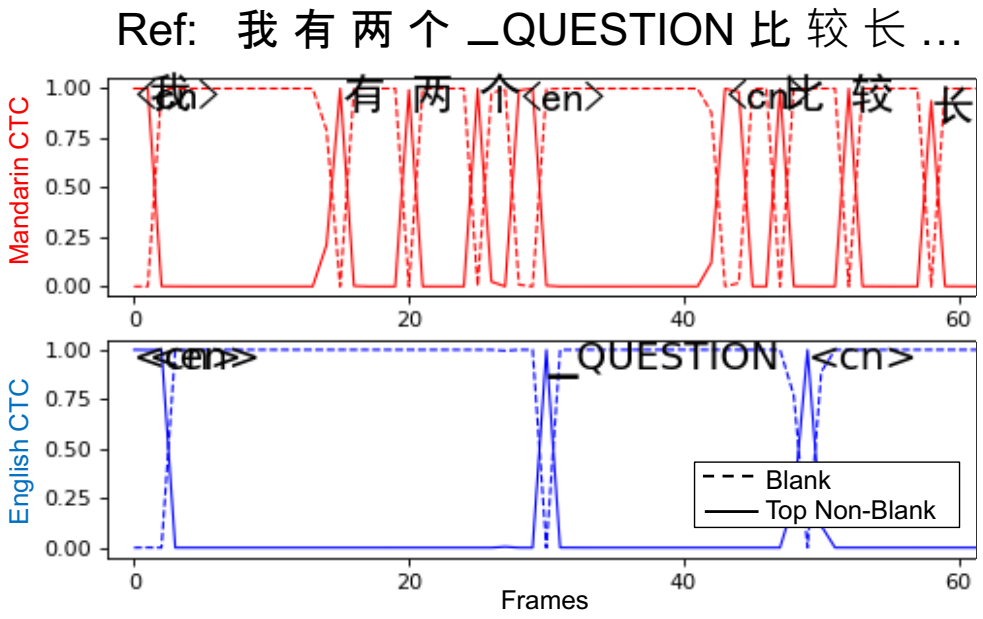

Figure 3.2: Example illustrating the label-to-frame posteriors of the monolingual CTC sub-nets for an intra-sententially CS utterance.

Language-Separation Ability of Monolingual Modules

Recall that in Eq. equation 3.2 we state that the bilingual Z is specified in terms of its monolingual parts, Z^M and Z^E, such that at any position t only one of z_t^M or z_t^E may be non-blank. If our proposed Conditional RNN-T model is indeed modeling $p(Z^M|X)$ and $p(Z^E|X)$ with this property, it logically follows that each monolingual CTC sub-net would be capable of emitting labels for frames corresponding to their language while emitting blanks for frames in the other language; we refer to this diarization-like (Lyu et al., 2013) ability as Language-Separation. This Language-Separation is observable in Figure 3.2 which depicts the blank and non-blank posterior values of each monolingual CTC sub-net for a snippet of CS speech. Here, the Mandarin side emits characters while the English side emits blanks except for frames 30 to 40 where the opposite occurs.

As mentioned in §3.3, there are two ways to optimize the Conditional RNN-T model towards the conditionally factorized formulation in Eq. equation 3.13. Table 3.2 shows that the implicit way, which uses pre-training to condition the bilingual task $p(Z|Z^M, Z^E)$, produces monolingual CTC sub-nets that reasonably able perform Language-Separation on CS data. However, the explicit way described in Eq. equation 3.14 is preferred since it provides a supervised Language-Separation signal. Thus, the resultant monolingual CTC sub-nets of this Conditional RNN-T + LS model have

Table 3.2: Ablation study examining the language-separation ability of the monolingual CTC sub-nets, $p(Z^M|X)$ and $p(Z^E|X)$, on the CS dev set. The sub-nets are expected to transcribe speech from their languages while ignoring speech in the other. Performance is evaluated using monolingual parts Y^M and Y^E of the ground-truth CS label sequence Y. CER/WER and Insertion Rate (INS) are shown.

	MAN PORTION OF CS			ENG PORTION OF CS		
Model	Sub-Net	CER	INS	Sub-Net	WER	INS
Cond. RNN-T	$p(Z^M\|X)$	11.8	3.7	$p(Z^E\|X)$	42.7	7.9
Cond. RNN-T + LS	$p(Z^M\|X)$	**8.6**	**0.7**	$p(Z^E\|X)$	**37.1**	**4.6**

Table 3.3: Experimental validation of conditional independence assumption in the bilingual module which models $p(Z|Z^M, Z^E, \not{X})$. The 3-Encoder variant removes this assumption in its bilingual module $p(Z|Z^M, Z^E, X)$. Results are shown on the CS dev set.

	Bilingual	CODE-SWITCHED		
Model	Condition	MER	CER	WER
Cond. RNN-T + LS	$p(Z\|Z^M, Z^E)$	**11.1**	**8.9**	**31.1**
3-Enc. RNN-T + LS	$p(Z\|Z^M, Z^E, X)$	11.2	9.0	**31.1**

a greater Language-Separation ability as indicated by the reduced insertion errors for the same CS data.

Conditional Independence of $p(Z|Z^M, Z^E)$ from X

Finally, we experimentally validate the conditional independence assumption in Eq. equation 3.5 that the final bilingual output Z depends only on monolingual alignment information Z^M and Z^E and nothing else from the observation X. In this study, we augment the Conditional RNN-T + LS model with a third ENCODER_A which maps the observation X to hidden representations $\mathbf{h}^A = \{\mathbf{h}_t^A \in \mathbb{R}^D | t = 1, ..., T\}$. This $\mathbf{h}^A$ is then added as a third term to the fusion in Eq. equation 3.9 thereby allowing the bilingual module to model $p(Z|Z^M, Z^E, X)$ instead of $p(Z|Z^M, Z^E)$. We call this modified model the 3-Encoder RNN-T + LS and train it in the same way as the Conditional RNN-T + LS model. As shown in Table 3.3, this 3-Encoder variant does not capture any additional useful information from the observation X but rather performs slightly worse than our Conditional RNN-T + LS model.

3.6 Conclusion

We present an end-to-end framework for jointly modeling CS and monolingual ASR with conditional factorization such that the bilingual task is logically decomposed into simpler sub-components. We show improvements on both CS and monolingual ASR over prior works, suggesting that our general joint modeling approach is promising towards building robust bilingual systems.

We've now examined the bilingual speech recognition, which requires parallel temporal multi-sequence modeling, under the setting where some amount of code-switched supervision is available. Next, we examine the resource constrained setting where no code-switched supervision is available.

Chapter 4

Application of *Align-then-Stitch* to Zero-Shot Code-Switched Speech Recognition

Summary

In this chapter, we seek to build effective code-switched (CS) automatic speech recognition systems (ASR) under the zero-shot setting where no transcribed CS speech data is available for training. The previously proposed *Align-then-Stitch* framework which conditionally factorizes the bilingual task into its constituent monolingual parts is a promising starting point. However, the method requires the monolingual modules to perform *language segmentation*. That is, each monolingual module has to simultaneously detect CS points and transcribe speech segments of one language while ignoring those of other languages – not a trivial task. We simplify each monolingual module by allowing them to transcribe all speech segments indiscriminately with a monolingual script (i.e. *transliteration*). This modification passes the responsibility of CS point detection to subsequent bilingual modules which determine the final output by considering multiple monolingual transliterations along with external language model information. We apply this transliteration-based approach in an end-to-end differentiable neural network and demonstrate its efficacy for zero-shot CS ASR on Mandarin-English SEAME test sets.

4.1 Introduction

In order to build multilingual automatic speech recognition (ASR) systems that are robust to code-switching (CS), practitioners must tackle both the long-tail of possible language pairs (Lewis,

2009) and the relative infrequency of intra-sententially CS examples within collected training corpora (Gambäck and Das, 2014). Therefore, a preeminent challenge in the CS ASR field is to build effective systems under the zero-shot setting where no CS ASR training data is available. Recent advancements in multilingual speech recognition have demonstrated the impressive scale of cross-lingual sharing in neural network approaches (Watanabe et al., 2017a; Li et al., 2019a, 2021; Yan et al., 2021; Lu et al., 2022; Bapna et al., 2022; Li et al., 2022b; Bai et al., 2022; Zhou et al., 2022a; Zhang et al., 2022b), and these works have shown that jointly modeling ASR with language identity (LID) grants some intra-sentential CS ability (Zhou et al., 2022a; Seki et al., 2018). However, most of these large scale models skew towards high-resourced languages (Li et al., 2022b) and do not seek to directly optimize for intra-sentential CS ASR between particular language pairs.

A more promising direction towards zero-shot CS ASR can be found in prior works which seek to incorporate monolingual data directly to improve CS performance (Gonen and Goldberg, 2018; Li et al., 2019b; Shan et al., 2019a; Taneja et al., 2019; Shi et al., 2020; Shah et al., 2020; Lu et al., 2020; Zhou et al., 2020b; Chuang et al., 2020; Dalmia et al., 2021a; Zhang et al., 2021a; Liu and Cao, 2021; Ali et al., 2021; Diwan et al., 2021b; Deng et al., 2022b). In particular, there are several works which achieve joint modeling of CS and monolingual ASR by conditionally factorizing the overall bilingual task into monolingual parts (Yan et al., 2022c; Tian et al., 2022; Song et al., 2022). By using label-to-frame synchronization, this *conditionally factorized* approach can make a CS prediction given only the predictions of the monolingual parts (Yan et al., 2022c) – theoretically these conditionally factorized models can model CS ASR without any CS data, but this has not been previously confirmed.

In this work, we seek to build CS ASR systems under two zero-shot data conditions: 1) monolingual speech and CS text data are available, 2) only monolingual speech and text data are available. In particular, we are interested in exploring the zero-shot capability of conditionally factorized joint CS and monolingual ASR models.

We first re-formulate the initial monolingual stage of these conditionally factorized models in terms of their *language segmentation* burden, showing that prior works expect each monolingual module to perform CS point detection and transcription in tandem. Any errors in CS point detection are thus propagated downstream to the final bilingual stage which attempts to stitch multiple monolingual predictions into an output which may or may not be CS. To improve model robustness towards zero-shot CS ASR, we propose an alternative formulation of the monolingual stage such that each module is an indiscriminate *transliterator*, transcribing all speech using a monolingual script without any regard for potential CS points. As a result we delay CS point detection until the final bilingual stage, allowing our models to condition this critical decision on multiple monolingual inputs and incorporate additional information from external language models. Our transliteration-based method yielded 5 absolute error-rate reduction in our zero-shot CS ASR experiments.

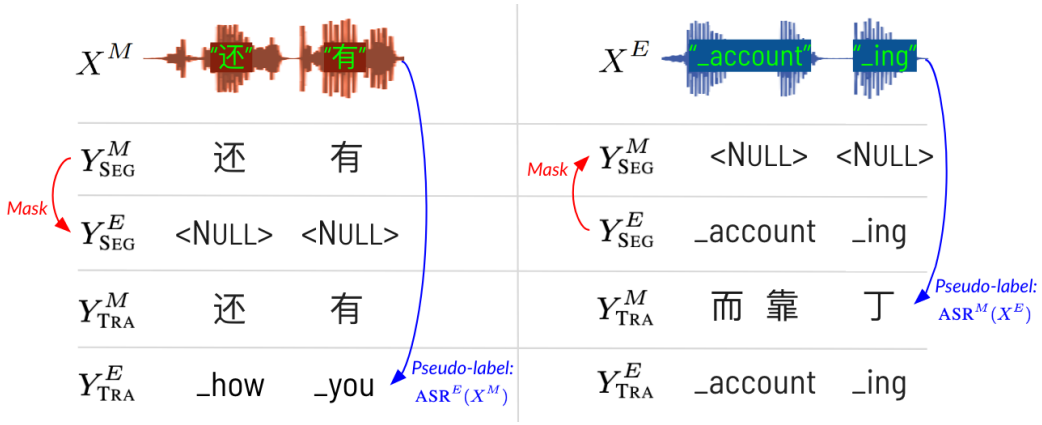

Figure 4.1: Examples showing the difference between language segmentation targets $Y^{M/E}_{\text{SEG}}$ obtained via masking (§4.2) vs. transliteration targets $Y^{M/E}_{\text{TRA}}$ obtained via cross-lingual pseudo-labeling (§4.3).

4.2 Background and Motivation

In this section, we examine the language segmentation role of the monolingual modules in previously proposed conditionally factorized models (Yan et al., 2022c), motivating our transliteration-based approach (§4.3).

Joint Modeling of Code-Switched and Monolingual ASR

Let us take the Mandarin-English bilingual pair as an example for the following formulations. Bilingual ASR, where speech may or may not be CS, is a sequence mapping from a T-length speech feature sequence, $X = \{\mathbf{x}_t \in \mathbb{R}^D | t = 1, ..., T\}$, to an L-length label sequence, $Y = \{y_l \in (\mathcal{V}^M \cup \mathcal{V}^E) | l = 1, ..., L\}$ consisting of Mandarin $\mathcal{V}^M$ and English $\mathcal{V}^E$. The conditionally factorized framework (Yan et al., 2022c) decomposes this bilingual task into three sub-tasks: 1) recognizing Mandarin, 2) recognizing English, and 3) composing recognized monolingual segments into a bilingual sequence.

The basis of this approach is to model the label-to-frame alignments. For each T-length observation sequence X and L-length bilingual label sequence Y there are a number of possible T-length label-to-frame sequences $Z = \{z_t \in \mathcal{V}^M \cup \mathcal{V}^E \cup \{\varnothing\} | t = 1 \dots T\}$, where $\varnothing$ denotes a blank symbol as in Connectionist Temporal Classification (CTC) (Graves et al., 2006) or RNN-T (Graves, 2012a). Further consider that for each bilingual Z there are two corresponding monolingual label-to-frame sequences $Z^M = \{z^M_t \in \mathcal{V}^M \cup \{\varnothing\} | t = 1 \dots T\}$ and $Z^E = \{z^E_t \in \mathcal{V}^E \cup \{\varnothing\} | t = 1 \dots T\}$. The label posterior, $p(Y|X)$, can thus be represented in terms of bilingual, $p(Z|X)$, and monolingual, $p(Z^M|X)$ and $p(Z^E|X)$, label-to-frame posteriors

as follows:

$$p(Y|X) = \sum_{Z \in \mathcal{Z}} \sum_{Z^M \in \mathcal{Z}^M} \sum_{Z^E \in \mathcal{Z}^E} p(Z, Z^M, Z^E|X) \tag{4.1}$$

where $\mathcal{Z}$ and $\mathcal{Z}^{M/E}$ denote sets of all possible bilingual and monolingual label-to-frame alignments for a given Y. Eq. equation 4.1 is the exact *joint* bilingual and monolingual ASR likelihood which can be further factorized using independence assumptions to obtain the form:

$$p(Y|X) \approx \underbrace{\sum_{Z} p(Z|Z^M, Z^E)}_{\text{Bilingual Posterior}} \underbrace{\sum_{Z^M} p(Z^M|X) \sum_{Z^E} p(Z^E|X)}_{\text{Monolingual Posteriors}} \tag{4.2}$$

From Eq. equation 4.1 to Eq. equation 4.2, the first assumption is that given Z^M and Z^E, no other information from the observation X is required to determine Z, allowing for conditional modeling of the bilingual posterior $p(Z|Z^M, Z^E, \cancel{X})$ given only monolingual information. The second assumption is that given X, Z^M and Z^E are independent, allowing for separate modeling of monolingual posteriors $p(Z^M|\cancel{Z^E}, X)$ and $p(Z^E|\cancel{Z^M}, X)$. Note we abbreviate this pair of separate monolingual modules as $p(Z^{M/E}|X)$ in future sections.

Modeling $p(Z^{M/E}|X)$ with Language Segmentation

What should be the behavior of the monolingual Mandarin module $p(Z^M|X)$ when encountering a segment of English speech and vice versa? Monolingual modules in prior works (Yan et al., 2022c; Tian et al., 2022; Song et al., 2022) determine each label-to-frame alignment $z_t^{M/E}$ by first determining the language identity of each speech frame $\text{LID}(\mathbf{x}_t)$ (Liu et al., 2021). If the speech frame $\mathbf{x}_t$ is from a foreign language then the module will ignore it by emitting a special NULL token, otherwise it will transcribe using its monolingual vocabulary. This monolingual *language segmentation* decision is defined as follows (shown for Mandarin):

$$z_t^M = \begin{cases} \text{argmax}_{m \in \mathcal{V}^M \cup \{\varnothing\}} \, p(z_t^M = m|X, z_{1:t-1}^M) & \text{if LID}(\mathbf{x}_t) \text{ is } M \\ \text{argmax}_{m \in \{\text{NULL}, \varnothing\}} \, p(z_t^M = m|X, z_{1:t-1}^M) & \text{if LID}(\mathbf{x}_t) \text{ is } E \end{cases} \tag{4.3}$$

Note that the frame-wise $\text{LID}(\mathbf{x}_t)$ is not a separate module, but rather an implicit decision within the posterior maximization over the NULL augmented monolingual label-to-frame alignments $Z^{M/E} = \{z_t^{M/E} \in \mathcal{V}^{M/E} \cup \{\varnothing, \text{NULL}\} | t = 1 \dots T\}$. This language segmentation behavior is

learned by optimizing likelihoods of NULL masked label targets Y^M_{SEG} and Y^E_{SEG} (e.g. in Figure 4.1).

It follows that the bilingual $p(Z|Z^M, Z^E)$ (Eq. equation 4.2) behaves as:

$$z_t = \begin{cases} m & \text{if } m \in \mathcal{V}^M \wedge e = \text{NULL} \\ e & \text{if } e \in \mathcal{V}^E \wedge m = \text{NULL} \\ b & \text{otherwise} \end{cases} \tag{4.4}$$

where m and e are the arguments maximizing $p(z^M_t|X, z^M_{1:t-1})$ and $p(z^E_t|X, z^E_{1:t-1})$ respectively and b is the argument maximizing $p(z_t|Z^M, Z^E, z_{1:t-1})$. If either monolingual module predicts a CS point by emitting NULL then the bilingual module defaults to the prediction of the other monolingual module – in other words, the first two cases of Eq. equation 4.4 expect that the language segmentation in Eq. equation 4.3 is mistake-free. The third fall-back case is considered for ambiguous language segmentation, such as if m and e are both NULL or both non NULL. This case-by-case bilingual decision is an adverse design for our zero-shot objective – models are likely to become over-reliant on the first two cases during training. Language segmentation while training on purely monolingual utterances boils down to an over-simplified *utterance-level* language identification task which may not generalize to *intra-sententially* CS test utterances. If CS point detection is expected to be tricky, then a more robust strategy should *always* expect ambiguous monolingual inputs to the final bilingual decision as in the third case of Eq. equation 4.4.

4.3 Proposed Framework

In this section, we propose to completely remove language segmentation from monolingual modules using a transliteration-based formulation of $p(Z^{M/E}|X)$. We then present a neural model of our modified conditionally factorized approach for zero-shot CS ASR.

Modeling $p(Z^{M/E}|X)$ with Transliteration

Rather than detecting CS points at the monolingual stage in order to know which speech segments to transcribe vs. which to ignore, we propose to simply allow each monolingual module to transcribe everything. This means that for speech of a foreign language the monolingual modules are producing *transliterations*, mapping sounds to phonetically similar units within their monolingual vocabularies $\mathcal{V}^M$ and $\mathcal{V}^E$. In other words, the monolingual modules simplify from Eq. equation 4.3 to the following form (shown for Mandarin):

$$z^M_t = \underset{m \in \mathcal{V}^M \cup \{\varnothing\}}{\text{argmax}} \; p(z^M_t = m|X, z^M_{1:t-1}) \tag{4.5}$$

where the speech X may contain any language. This form completely removes any sense of frame-wise language identity $\text{LID}(\mathbf{x}_t)$.

To see why this modification is advantageous for zero-shot CS ASR, consider the corresponding change to the bilingual module:

$$z_t = \underset{b \in \mathcal{V}^M \cup \mathcal{V}^E \cup \{\varnothing\}}{\text{argmax}} \; p(z_t = b | Z^M, Z^E, z_{1:t-1}) \tag{4.6}$$

Note that this new bilingual form in Eq. equation 4.6 never defaults to the prediction of one monolingual module as in the first two cases of the previously proposed bilingual form in Eq. equation 4.4, reducing the risk of propagating errors made in the monolingual stage. In other words, the bilingual decision now determines each z_t by directly considering the conditional likelihood $p(z_t | Z^M, Z^E, z_{1:t-1})$ (Eq. equation 4.2). This modification effectively delays CS point detection from the monolingual stage (where we would have to simultaneously transcribe and perform frame-wise language identification per §4.2), to the bilingual stage (where transcription information is already given).

To train monolingual modules to transliterate speech segments of a foreign language, we obtain transliteration targets Y^M_{TRA} and Y^E_{TRA} using cross-lingual pseudo-labeling.[1] For instance, we pass monolingual English speech X^M to a monolingual Mandarin ASR model $\text{ASR}^M(\cdot)$ for inference and vice versa as follows (e.g. in Figure 4.1):

$$Y^M_{\text{TRA}} \leftarrow \text{ASR}^M(X^E) \tag{4.7}$$

$$Y^E_{\text{TRA}} \leftarrow \text{ASR}^E(X^M) \tag{4.8}$$

where $\text{ASR}^{M/E}(\cdot)$ denote generic label-to-frame models – if we use the same architecture for pseudo-labeling as we do for our monolingual modules then these transliteration targets are cross-lingual semi-supervisions (Jyothi and Hasegawa-Johnson, 2015; Thomas et al., 2020; Billa, 2021; Lugosch et al., 2022).[2] Swapping the language segmentation targets Y^M_{SEG} and Y^E_{SEG} (§4.2) for these transliteration targets Y^M_{TRA} and Y^E_{TRA} is the *only* modification required to realize our desired monolingual and bilingual module behaviors in Eq. equation 4.5 and equation 4.6.

Conditional CTC with External LM Architecture

Finally, let us consider how to construct a neural architecture for our modified conditionally factorized framework. Monolingual and bilingual label-to-frame posteriors (§4.2) may be modeled using

[1] Unlike text-based transliteration (Knight and Graehl, 1998), pseudo-labeling relies solely on the resources presumed to be available in our zero-shot CS ASR settings.

[2] We can apply transliteration to CS speech by stitching predictions corresponding to forced aligned (Kürzinger et al., 2020) foreign segments between true native targets.

CTC or RNN-T networks as demonstrated by prior works (Yan et al., 2022c; Tian et al., 2022; Song et al., 2022). However for zero-shot CS ASR, the conditional independence assumption of CTC vs. the internal language modeling of RNN-T is a critical difference. A RNN-T based model may require internal language model (LM) adaptation (Graves, 2012a; Meng et al., 2021; Zhou et al., 2022b) to alleviate monolingual biases while a CTC based model can be directly applied to CS test sets with optional shallow external LM fusion (Hannun et al., 2014).

We therefore model monolingual, $p(Z^M|X)$ and $p(Z^E|X)$, and bilingual likelihoods, $p(Z|Z^M, Z^E)$, using CTC networks, $P_{\text{M\_CTC}}(\cdot)$, $P_{\text{E\_CTC}}(\cdot)$, and $P_{\text{B\_CTC}}(\cdot)$, as follows:

$$P_{\text{M\_CTC}}(z_t^M|X, \cancel{z_{1:t-1}^M}) = \text{SOFTMAXOUT}^M(\mathbf{h}_t^M) \tag{4.9}$$

$$P_{\text{E\_CTC}}(z_t^E|X, \cancel{z_{1:t-1}^E}) = \text{SOFTMAXOUT}^E(\mathbf{h}_t^E) \tag{4.10}$$

$$P_{\text{B\_CTC}}(z_t|\mathbf{h}^M, \mathbf{h}^E, \cancel{z_{1:t-1}}) = \text{SOFTMAXOUT}^B(\mathbf{h}_t^M + \mathbf{h}_t^E) \tag{4.11}$$

where speech encoders, ENCODER^M and ENCODER^E, map the speech signal, X, to latent monolingual representations, $\mathbf{h}^M = \{\mathbf{h}_t^M \in \mathbb{R}^D | t = 1, ..., T\}$ and $\mathbf{h}^E = \{\mathbf{h}_t^E \in \mathbb{R}^D | t = 1, ..., T\}$ followed by softmax normalized linear projections to monolingual or bilingual vocabularies. Then addition fusion yields a bilingual latent representation which is finally fed to the bilingual CTC. These three CTC networks are jointly optimized with an interpolated multi-task objective: $\mathcal{L} = \lambda_1 \mathcal{L}_{\text{B\_CTC}} + (1 - \lambda_1)(\mathcal{L}_{\text{M\_CTC}} + \mathcal{L}_{\text{E\_CTC}})/2$.

During decoding, we first merge all CTC likelihoods, $P_{\text{M\_CTC}}(\cdot)$, $P_{\text{E\_CTC}}(\cdot)$, and $P_{\text{B\_CTC}}(\cdot)$, following the interpolation procedure described in Eq. (6) of (Song et al., 2022); we denote this merged CTC likelihood as $P_{\text{CTC}}(Z|X)$. We then jointly decode $P_{\text{CTC}}(\cdot)$ with an external bilingual LM, $P_{\text{B\_LM}}(Y)$, using the time-synchronous beam search described in (Hannun et al., 2014), which approximates the following decision:

$$\underset{Y \in \{\mathcal{V}^M \cup \mathcal{V}^E\}*}{\text{argmax}} \lambda_2 (\prod_{Z \in \mathcal{Z}} \log P_{\text{CTC}}(\cdot)) + (1 - \lambda_2) \log P_{\text{B\_LM}}(\cdot) \tag{4.12}$$

where $\{\mathcal{V}^M \cup \mathcal{V}^E\}*$ denotes the set of all possible bilingual outputs.[3] This architecture, which we refer to as Conditional CTC, is depicted by the block-diagram in Figure 4.2. The monolingual modules of these Conditional CTC models can perform either language segmentation (§4.2) or transliteration (§4.3) depending on which set of monolingual targets (e.g. Figure 4.1) is used during training. For transliteration, we obtain Y_{TRA}^M and Y_{TRA}^E (Eq. equation 4.7 and equation 4.8) by greedily decoding monolingual CTC models (Eq. equation 4.9 and equation 4.10) and then applying repeat and blank removal.

[3]For language segmentation variants of Conditional CTC, we do not expand hypotheses with the special NULL token to avoid corrupt outputs.

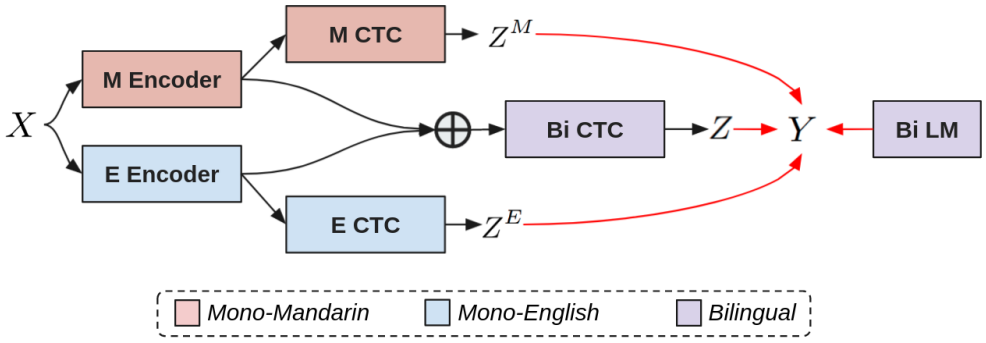

Figure 4.2: Conditional CTC architecture consisting of monolingual and bilingual CTC's plus an external bilingual LM. Red lines indicate joint decoding via time-synchronous beam search.

4.4 Data and Experimental Setup

Data: We apply 0.9 and 1.1 speed perturbations to up-sample SEAME (Lyu et al., 2010) training data by 3x, resulting in 291h of total labeled speech data. We then split the training data into CS and monolingual (Mandarin + English) parts to create two zero-shot settings. The first setting allows 87h of monolingual labeled speech data (for ASR training) and 89k lines of unpaired CS or monolingual text data (for LM training). The second fully zero-shot setting further removes the CS unpaired text data, leaving 39k lines of unpaired monolingual text data. Both settings remove 204h of CS labeled speech data. Monolingual CTC's trained on the English and Mandarin only SEAME splits were used for cross-lingual pseudo-labeling (§4.3).
Models: Models are trained using ESPnet (Watanabe et al., 2018). We combine 4000 Mandarin characters with 4000 English BPE (Sennrich et al., 2016) units to form the output vocabulary. Conditional CTC models have 2 conformer encoders (Gulati et al., 2020; Guo et al., 2021) with 12 blocks, 4 heads, 15 kernel size, 2048 feed-forward dim, 256 and attention dim. Vanilla CTC baselines with only 1 encoder use 512 attention dim, so all models have about 80M parameters. All models are initialized with encoder(s) pre-trained on 150h of Mandarin AISHELL-1 (Bu et al., 2017) and/or 118h of English TED-LIUM-v1 (Rousseau et al., 2012). We set $\lambda_1 = 0.7$ (§4.3) during training for 40 epochs. We set $\lambda_2 = 0.8$ (§4.3) during decoding with beam 10. We use RNN-LMs with 4 layers and 2048 dim trained for 20 epochs.
Evaluation: Systems are evaluated on the full SEAME test sets (devman and devsge) and also scored individually on the CS and monolingual portions of these sets. We measure mixed error-rate (MER) that considers word-level English and character-level Mandarin.

Table 4.1: Results comparing Conditional CTC models with *transliteration*-based monolingual modules to their *language segmentation* counterparts and Vanilla CTC baselines. The 1st horizontal partition shows top-line results when CS ASR training data is available. The 2nd and 3rd partitions show zero-shot results when only monolingual ASR training data is available. Performances on the full, CS only, and monolingual only splits of the SEAME test sets are measured by % mixed error rate (MER ↓). All models use CTC + LM decoding.

			ASR	LM	DEVMAN			DEVSGE		
ID	Model	Monolingual Behavior	Data	Data	Full	CS	M	Full	CS	M
A1	Vanilla CTC	*No Monolingual Modules*	CS + M	CS + M	18.8	18.2	21.5	26.2	23.7	29.8
A2	Conditional CTC	Language Segmentation	CS + M	CS + M	**17.1**	**16.5**	19.9	**23.5**	**21.4**	**26.5**
A3	Conditional CTC (Ours)	Transliteration	CS + M	CS + M	17.3	16.9	**19.1**	24.0	22.1	26.7
B1	Vanilla CTC	*No Monolingual Modules*	M	CS + M	36.6	38.9	27.0	42.5	47.0	36.1
B2	Conditional CTC	Language Segmentation	M	CS + M	30.1	32.0	22.0	35.7	39.7	**30.1**
B3	Conditional CTC (Ours)	Transliteration	M	CS + M	**25.2**	**26.0**	**21.9**	**31.0**	**31.5**	30.2
C1	Vanilla CTC	*No Monolingual Modules*	M	M	39.1	41.6	28.4	44.8	50.0	37.3
C2	Conditional CTC	Language Segmentation	M	M	32.2	34.4	23.0	37.8	42.6	31.1
C3	Conditional CTC (Ours)	Transliteration	M	M	**27.3**	**28.5**	**22.6**	**32.7**	**34.0**	**30.8**

4.5 Results

Table 4.1 presents results in three horizontal partitions where 1) all SEAME training data is allowed 2) CS speech data is removed and 3) CS speech and text data are removed; the latter two settings emulate practical zero-shot scenarios. When CS speech data is available, language segmentation is reliable and thus the transliteration-based method is not necessary (A2 vs. A3). However, once CS speech data is removed the language segmentation approach degrades 13 absolute MER on both full test sets; as a result the transliteration approach outperforms by 5 absolute MER, a wide margin, owing primarily to superior performance on CS utterances (B2 vs. B3). When CS text data is also removed both variants of Conditional CTC degrade only by an additional 2 absolute MER and the gap between remains (C2 vs. C3). In all three data settings both Conditional CTC models outperform Vanilla CTC baselines.

Ablations on the Conditional CTC Model

Our Conditional CTC models consist of three types of modules: monolingual CTC's ($P_{\text{M\_CTC}}$ and $P_{\text{E\_CTC}}$), bilingual CTC ($P_{\text{B\_CTC}}$), and bilingual LM ($P_{\text{B\_LM}}$). In Table 4.2, we examine the relative contributions of these modules by removing each from model B3 of Table 4.1 during joint decoding (described in §4.3). Removing the bilingual LM (line 2) degrades performance more than removing the bilingual CTC (line 5), showing the importance of utilizing CS textual data

Table 4.2: Ablation study examining the relative importance of monolingual CTC, bilingual CTC, and bilingual LM modules during decoding as measured by % mixed error rate (MER ↓) on the devman test set. Bilingual modules are shown in blue and the most severely degraded combination (with no bilingual modules) is **bolded**.

#	Model	Decoding Likelihoods	MER(↓)
1	Cond. CTC w/ Trans.	$P_{\text{M\_CTC}}$, $P_{\text{E\_CTC}}$, $P_{\text{B\_CTC}}$, $P_{\text{B\_LM}}$	25.2
2	− Bilingual LM	$P_{\text{M\_CTC}}$, $P_{\text{E\_CTC}}$, $P_{\text{B\_CTC}}$	27.4
3	− Monolingual CTCs	$P_{\text{B\_CTC}}$, $P_{\text{B\_LM}}$	25.7
4	− Bilingual LM	$P_{\text{B\_CTC}}$	27.9
5	− Bilingual CTC	$P_{\text{M\_CTC}}$, $P_{\text{E\_CTC}}$, $P_{\text{B\_LM}}$	26.0
6	− Bilingual LM	$P_{\text{M\_CTC}}$, $P_{\text{E\_CTC}}$	**48.1**

when available. Further, note that monolingual CTCs do contribute (line 3), but are insufficient on their own (line 6). Finally, the fact performance is still reasonable without the bilingual CTC (line 5) suggests that separately trained monolingual CTCs may be directly applied to CS ASR if a CS LM is available – this direction may offer a high degree of scalability towards the long-tail of possible CS pairs and towards CS between three or more languages.

Relaxing the Zero-Shot Setting

How much CS ASR training data do we need for the originally proposed language segmentation method (§4.2) to be sufficient? The answer depends on the proximity of the particular language pair and characteristics of the dataset being used, but in our experimental setup we find that the answer is 2h of CS speech data (see Figure 4.3). The decreasing effectiveness of our transliteration method for increasing amounts of CS ASR training data suggests that the cross-lingual pseudo-labels are noisy to a degree. Future investigations into improving pseudo-labeling quality (e.g. via constrained decoding) may benefit this work and other related techniques which employ cross-lingual semi-supervision (Jyothi and Hasegawa-Johnson, 2015; Thomas et al., 2020; Billa, 2021; Lugosch et al., 2022).

4.6 Conclusion

We identify that the promising conditionally factorized joint CS and monolingual ASR framework has an acute weakness which limits its applicability to zero-shot CS ASR; the original formulation expects that each monolingual module can cleanly transcribe native speech while ignoring foreign

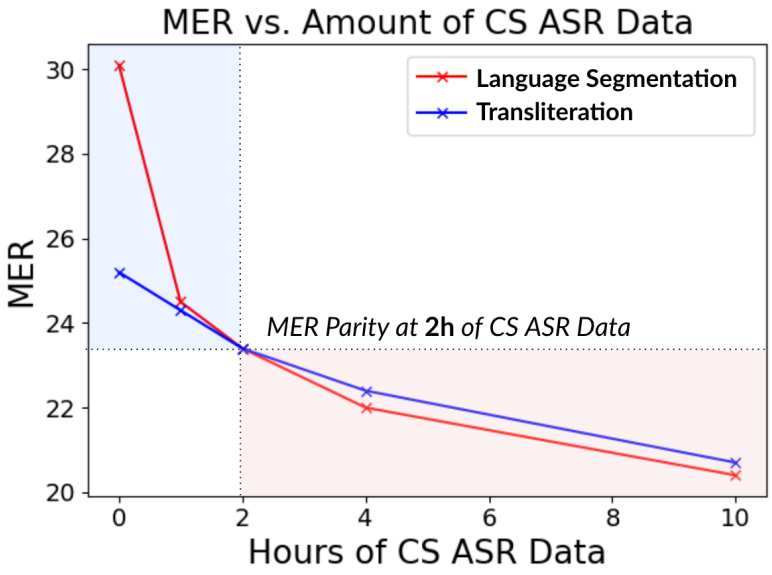

Figure 4.3: Analysis on the amount of CS ASR training data required for conditional CTC with language segmentation to outperform conditional CTC with transliteration. MER(↓) on devman is shown.

speech. We propose a simple modification via cross-lingual pseudo-labeling to allow the monolingual modules to instead produce transliterations of foreign speech, thereby avoiding error propagation of frame-wise LID decisions. We demonstrate the effectiveness of our transliteration-based method using Conditional CTC models deployed for zero-shot Mandarin-English CS ASR. In future work, we will extend to other languages, scale beyond bilingualism, and refine our pseudo-labeling technique.

We've now established the efficacy of parallel, temporal multi-sequence modeling for bilingual and code-switched speech recognition. Next, we examine code-switching on a larger scale of languages.

Chapter 5

CS-FLEURS: 100 Language Pair Code-Switched Speech Recognition Benchmark

Summary

In this chapter we present CS-FLEURS, a new dataset for developing and evaluating code-switched speech recognition and translation systems beyond high-resourced languages. CS-FLEURS consists of 4 test sets which cover in total 113 unique code-switched language pairs across 52 languages: 1) a 14 X-English language pair set with real voices reading synthetically generated code-switched sentences, 2) a 16 X-English language pair set with generative text-to-speech 3) a 60 {Arabic, Mandarin, Hindi, Spanish}-X language pair set with the generative text-to-speech, and 4) a 45 X-English lower-resourced language pair test set with concatenative text-to-speech. Besides the four test sets, CS-FLEURS also provides a training set with 128 hours of generative text-to-speech data across 16 X-English language pairs. Our hope is that CS-FLEURS helps to broaden the scope of future code-switched speech research.

5.1 Introduction

In the current era of massively multilingual speech processing, practitioners are developing and evaluating systems on hundreds of languages (Li et al., 2022b; Radford et al., 2023; Zhang et al., 2023; Peng et al., 2024; Pratap et al., 2024). This increasingly global impact is due in large part to dataset construction efforts, which have enabled pre-training on raw untranscribed speech (Kahn et al., 2020; Wang et al., 2021) and supervised training on transcribed or pseudo-labeled speech

(Panayotov et al., 2015; Pratap et al., 2020b; Ardila et al., 2020; Li et al., 2023b), as well as broad benchmarking on standardized evaluation sets across many languages (Shi et al., 2023; Conneau et al., 2023). However, these data resources are collections of predominately *monolingual* utterances - we currently lack broad *code-switched* speech data, where utterances consist of more than one language.

Despite its prevalence in multilingual communities, conversational code-switched speech is particularly challenging to collect at scale. Unlike monolingual data, which can be sourced from a wide range of speakers, code-switched speech requires bilingual or multilingual speakers who naturally mix languages in conversation (Auer, 2013). Additionally, code-switching tends to occur sporadically and unpredictably, making it difficult to capture in large, curated datasets. Even when data is available, standardizing it across language pairs presents another challenge. These factors make it difficult to create a single, broadly representative dataset of code-switched speech; instead, practitioners have focused on constructing dedicated corpora for their particular communities of interest (Deuchar et al., 2014; Lyu et al., 2010; van der Westhuizen and Niesler, 2018; Chowdhury et al., 2021; Diwan et al., 2021a; Hamed et al., 2022; Ugan et al., 2024).

In this work, we take an alternative approach: rather than relying on the collection of natural conversational code-switched speech, we use a combination of read speech and synthetic speech over synthetically generated code-switched text. Although our approach does not capture code-switched speech that occurs in conversation, we are able to use (1) **standard code-switching patterns**, (2) a **standard textual domain**, and (3) a **standard audio domain** across language pairs. By controlling for these naturally occurring variations when constructing the dataset, we can then ask: *how do models perform on code-switched speech across different language pairs*?

We present CS-FLEURS: a massively multilingual and code-switched ASR and ST dataset consisting of 52 languages and 113 unique code-switched pairs across three subsets:

- CS-FLEURS-READ: 14 X-English pairs, read speech
- CS-FLEURS-XTTS: 76 pairs across 17 langs, generative TTS
- CS-FLEURS-MMS: 45 X-English pairs, concatenative TTS

To the best of our knowledge, this is the broadest single collection of code-switched speech data both in terms of languages covered and number of unique code-switched pairs.

This dataset supports model benchmarking across 113 diverse pairs and model training across 16 X-English pairs. Figure 5.1 illustrates the language coverage and Table 5.1 summarizes the dataset with comparison to other code-switched corpora. In addition to the dataset, this work also describes several empirical findings. First, we show that Whisper (Radford et al., 2023) struggles with transcribing code-switched speech, especially for language pairs with distinct scripts. However, we also show that Whisper is relatively robust at translating code-switched speech to English,

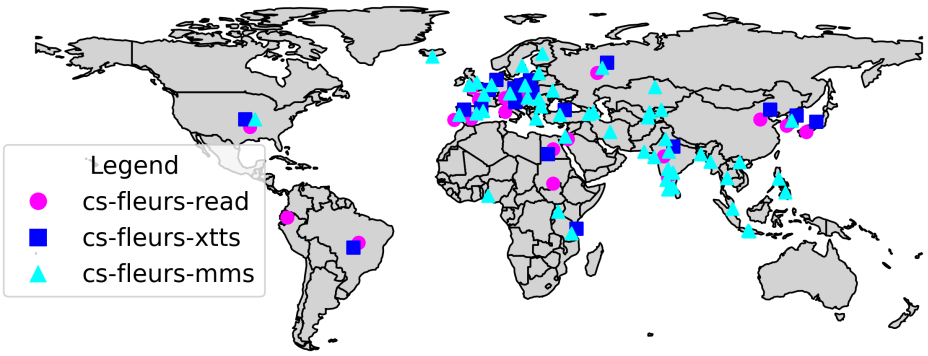

Figure 5.1: Illustration of where the 52 languages covered by CS-FLEURS are spoken around the world.

Table 5.1: CS-FLEURS vs other code-switched speech corpora.

Dataset	Langs	CS-Pairs	Domain	Speech Type	Hours
ArzEn (Hamed et al., 2022)	2	1	Conversation	Spontaneous	12
Bangor Miami (Deuchar et al., 2014)	2	1	Conversation	Spontaneous	35
SEAME (Lyu et al., 2010)	2	1	Conversation	Spontaneous	192
ESCWA (Chowdhury et al., 2021)	3	2	Conversation	Spontaneous	3
MUCS (Diwan et al., 2021a)	3	2	Lecture	Structured	160
Soapies (van der Westhuizen and Niesler, 2018)	5	4	TV	Scripted	14
CS-FLEURS	**52**	**113**	**Wikipedia**	**Read/Synthetic**	**294**

suggesting that code-switched ST is an easier task than ASR. Finally, we show that synthesizing code-switched speech data is an effective form of training data augmentation, improving model performance on both seen and unseen language pairs.

5.2 Dataset

CS-FLEURS builds upon a line of prior works. The FLoRes-101 dataset (Goyal et al., 2022) consists of 3001 sentences from English Wikipedia translated in 101 languages by human translators. Then, the FLEURS dataset (Conneau et al., 2023) took 2009 of those sentences from FLoReS-101 and recorded read speech in 102 languages with three speakers per language. Now, CS-FLEURS takes the 2009 utterances from FLEURS to generate code-switched text and speech across 113 language pairs, keeping the same train/dev/test splits from FLEURS.

Table 5.2: The 113 code-switched language pairs covered.

CS-Pairs	Matrix	Embedded	Read	XTTS	MMS
7	ara deu fra hin por rus spa	eng	✓	✓	✓
5	ces cmn ita jpn kor	eng	✓	✓	
2	slk tel	eng	✓		
3	nld pol tur	eng		✓	✓
1	hun	eng		✓	
60	ara cmn hin spa	ara fra deu ita por pol tur rus nld ces spa cmn jpn hun kor hin		✓	
35	ben bul cat ceb cym ell fin guj heb hun ind isl jav kan kaz kir lav lug mal mar mya pan ron swe swh tam tel tgk tgl tha ukr urd uzb yor zlm	eng			✓

We follow the Matrix Language-Frame model (Myers-Scotton, 1997); for each code-switched pair, one language, referred to as Matrix, provides grammatical structure while a second language, referred to as Embedded, provides words or morphological units that are inserted within the sentence. We refer to language pairs as Matrix-Embedded; for instance, Mandarin-English refers to a Mandarin matrix sentence with English words embedded. Table 5.2 shows a breakdown of the code-switched language pairs covered by CS-FLEURS across the CS-FLEURS-READ, CS-FLEURS-XTTS, and CS-FLEURS-MMS subsets. Table 5.3 provides additional summary statistics.

Table 5.3: Summary statistics for CS-FLEURS.

	READ	XTTS				MMS
Statistic	Test	Train	Dev	Test1	Test2	Test
Duration (hours)	17	128	15	36	42	56
Tokens (words)	128k	889k	105k	257k	300k	315k
Matrix Langs	14	16	16	16	4	45
Embedded Langs	1	1	1	1	15	1
Total CS Pairs	14	16	16	16	60	45
Same-Script Pairs	7	10	10	10	9	22
Distinct-Script Pairs	7	6	6	6	51	23

Collecting Read Speech Across 14 Language Pairs

CS-FLEURS-READ consists of read speech from bilingual speakers across 14 X-English language pairs (1-4 speakers per language pair, 2:1 M:F ratio overall). CS-FLEURS-READ is generated using the test set of FLEURS and is intended for model benchmarking. As such, this set has been human validated. In terms of language coverage, the limiting factor for constructing this set was the availability of bilingual readers. Finally, in this set we regard the 14 non-English languages as Matrix and English as Embedded.

Generating accurate and fluent code-switched text

We use a large language model (LLM) backbone for generating code-switched text for CS-FLEURS-READ. Specifically, by utilizing X-English parallel monolingual sentences from FLEURS test set, we prompt GPT-4o (Hurst et al., 2024) to generate the equivalent code-switched sentences in three different manners:

- *GPT-Base*: feeding only the paired monolingual sentences
- *GPT-EC*: feeding paired monolingual sentences and a set of valid English words to be embedded, determined by the Equivalence Constraint theory (Poplack, 1980) so as not to violate the syntactic rules of both languages, as implemented in (Kuwanto et al., 2024)
- *GPT-Pred*: feeding paired monolingual sentences and a set of valid English words to be embedded, determined by a predictive model (Hamed et al., 2023), which given an English sentence, predicts the plausible English words to be embedded in a corresponding code-switched sentence

Table 5.4: The prompt consists of three main parts: (1) task description and the paired monolingual sentences to be synthesized (outlined in black); (2) the set of valid English words in the case of *GPT-EC* and *GPT-Pred* (outlined in blue); and (3) morphological code-switching example in the case of languages where this type of switching is common (outlined in purple).

You are a bilingual Spanish-English speaker, you will help translate Spanish and English sentences into a code-mixed sentence. The code-mixed sentence should contain words from both languages, each written in the correct language script. We will provide you with both Spanish and English sentences in addition to some English keywords that need to be present in the produced code-mixed sentence. You need to produce morphological code-switching when appropriate to achieve better fluency. The following is an example: Spanish: Mi amigo se enojó cuando le revelaron el final de la película. English: My friend freaked out when someone spoiled the end of the movie. English keywords: freak out, spoiled A fluent code-mixed sentence would be: Mi amigo se frickeo cuando le spoilearon el final de la película. Please produce the code-mixed sentence for the following: Spanish sentence: {Spanish_sentence} English sentence: {English_sentence} English keywords: {English_keywords} Only provide the code-mixed sentence without explanation or extra text.

Table 5.4 shows an example of a full prompt for a given sentence pair. *GPT-EC* and *GPT-Pred* prompts include English keywords for embedding into the matrix language. Further, to encourage morphological code-switching, where more than one language occurs *within the same word*, we prompt GPT-4o with an in-context example, as illustrated in the table. 10 out of 14 languages were prompted as such.[1]

The resultant generations are not guaranteed to be (1) code-switched, (2) accurate, preserving the meaning of the original monolingual sentences, or (3) fluent, plausibly produced by a human. We ask our bilingual readers to reject sentences which are not code-switched or not accurate. Sentences are not rejected on fluency, but we collect ratings from 0-2, with 0 indicating unnaturalness, 1 indicating somewhat natural, and 2 indicating perfect naturalness.

As shown in Table 5.5, the reject rate was fairly consistent across the different prompts. In terms of fluency, *GPT-Pred* scored highest while *GPT-EC* was lowest; this result correlates with the frequency of switching, as measured by the Code-Mixing Index (CMI) (Gambäck and Das, 2014). We also noted that *GPT-EC* embeds function words a high rate, which bilingual readers found to be rather unnatural. *GPT-Pred* on the other hand primarily embeds content words, which resulted in over over 80% of generations to be rated somewhat or perfectly natural.

[1] 10 w/ morphological example: ara, ces, deu, ita, kor, por, rus, slk, spa, tel, tam; 4 w/o: cmn, fra, hin, jpn

Table 5.5: Human validation on LLM-generated code-switched text across 14 X-English pairs. CMI=Code-Mixing Index.

Prompt	Reject(%)	Fluency 0(%)	Fluency 1(%)	Fluency 2(%)	Fluency Avg	CMI
GPT-Base	**8.54**	31.66	25.44	34.36	1.03	25.50
GPT-EC	12.15	44.71	21.22	21.92	0.74	30.98
GPT-Pred	11.76	19.26	25.25	43.73	**1.28**	21.05

Reading and recording code-switched speech

To facilitate the collection of read speech, we developed an open-source toolkit.[2] We distributed generated code-switched text to 21 bilingual readers. For each sentence, we randomly selected one of the three aforementioned prompts, standardizing the selection across all language pairs. The bilingual readers were provided with the monolingual references along with the code-switched sentences. They were instructed to 1) verify that the sentence is indeed code-switched, 2) the sentence has the same general meaning as the monolingual reference, 3) read and record the sentence, and 4) rate the fluency of the sentence. Recordings were then validated by a human listener.

Synthesizing Speech Across 111 Language Pairs

Next, we describe two synthetic speech sets designed to complement CS-FLEURS-READ by covering additional language pairs and providing code-switched training data.

CS-FLEURS-XTTS consists of synthetic speech from the XTTS-v2 model (Casanova et al., 2024), which supports 17 languages, across 16 X-English language pairs as well as 60 non-English language pairs (15 Arabic-X, 15 Hindi-X, 15 Mandarin-X, 15 Spanish-X). For the 16 X-English language pairs, we generate train, dev, and test sets, regarding the 16 non-English languages as matrix and English as embedded — 12 of these language pairs overlap with CS-FLEURS-READ. The 60 non-English language pairs are test-only and we regard Arabic, Hindi, Mandarin, or Spanish as the matrix and the other language as embedded — none of these language pairs overlap with the other two subsets.

CS-FLEURS-MMS consists of synthetic speech from MMS TTS models (Pratap et al., 2024) across 45 X-English language pairs, many of which are lower-resourced — only 10 of these language pairs overlap with the other two subsets. This set is test-only and we regard the non-English languages as matrix.

[2]Toolkit reference redacted for anonymity

Table 5.6: Comparison of the code-switching frequency, as measured by Code-Mixing Index (CMI), of text generated by LLM prompts and align-then-swap (Swap) across 12 X-English language pairs. M=Mean; SD=Standard Deviation.

Type	ara	ces	cmn	deu	fra	hin	ita	jpn	kor	por	rus	spa	M	SD
LLM	12.3	30.4	24.1	27.8	36.0	18.5	37.0	18.5	7.0	37.2	17.7	39.8	25.5	10.8
Swap	20.4	20.1	14.11	16.8	15.0	13.6	16.8	11.2	25.9	17.0	20.6	15.5	17.2	**4.0**

Both sets are not human validated due to lack of bilingual speakers. Instead, we use language-universal forced alignment to filter low quality generations.

Cheaply and flexibly generating code-switched text

While the LLM backbone described in §5.2 can generate natural and morphologically rich code-switching, there are also a number of drawbacks which limit scalability aside from the high computational (or API) cost. The LLM-based methods required human validation to rejected around 10% of the sentences across 14 X-English language pairs; it is reasonable to expect a higher reject rate for lower resourced languages.

Further, we found that the LLM-based methods produced a wide range of code-switching frequency, as measured by CMI (Gambäck and Das, 2014), across language pairs. This variance is shown in the first line of Table 5.6: the standard deviation of the CMI across 12 language pairs in the human validated CS-FLEURS-READ is 10.8. We therefore opt for a more simple and rigid method for generating the code-switched text in CS-FLEURS-XTTS and CS-FLEURS-MMS, referred to as *align-then-swap*.

The align-then-swap procedure is simple. We first obtain word-level alignment using AwesomeAlign (Dou and Neubig, 2021), a 104 language mBERT model (Devlin et al., 2019) fine-tuned towards word alignment objectives using parallel text across five language pairs. Notably, AwesomeAlign generalizes to other languages that were part of the mBERT pre-training but unseen during the word alignment fine-tuning stage (Dou and Neubig, 2021). Next we randomly select 30% nouns, verbs, abverbs, and adjectives, tagged using Stanza (Qi et al., 2020), to be swapped with the aligned embedded language words.

Additionally, we take care of one-to-many word alignments (matrix-to-embedded) by inserting words in the order that they originally appeared in the monolingual embedded language sentence. We also take care of character-based languages (Mandarin and Japanese) by first re-segmenting to words using Stanza (Qi et al., 2020); Korean text in FLEURS was already word segmented, so this extra step was not necessary.

As shown in Table 5.6, the resultant CMI from align-then-swap (Swap) is more consistent across language pairs than its LLM-based counterpart: the standard deviation is only 4.0, which is 60% lower. This simpler, yet more consistent, method of generating code-switched text is therefore preferable for the long tail of lower resourced languages and rare language pairs.

Generative code-switched synthesis with XTTS

XTTS-v2 (Casanova et al., 2024) is a multilingually trained TTS model based on a GPT-2 encoder that predicts VQ-VAE units which then latently condition a HiFi-GAN decoder. The model was trained on 17 languages, which are distinguished by appending a language ID token (e.g. [en]) at the start of textual inputs. XTTS-v2 also incorporates a voice conversion component by conditioning both the GPT-2 encoder and VQ-VAE decoder on speaker information; in practice only single utterance from the target speaker is necessary to enable voice conversion.

To synthesize code-switched speech using XTTS-v2, we feed generated code-switched text to the model with the matrix language ID token.[3] Mandarin, Japanese, and Korean text is romanized, in the same manner as the original XTTS-v2 training. We use a matrix language speaker from the FLEURS dataset for the voice conversion component. XTTS-v2 produces 24khz speech, which we then down-sample to 16khz.

Concatenative code-switched synthesis with MMS-TTS

To cover additional language pairs, we use a concatenative approach that generates monolingual TTS in segments using MMS-TTS (Pratap et al., 2024). We concatenate the monolingual segments, inserting 100ms of silence in between. Since MMS-TTS is a single-speaker VITS style model, the resulting concatenated code-switched speech contains unnatural artifacts and speaker changes; however the content is correct. Unlike the XTTS-v2 outputs, MMS-TTS outputs do not contain any accented speech.

Universal forced alignment filtering

To perform quality control, we use a universal forced alignment model, MMS-ZS (Zhao et al., 2024), which was trained to align speech in thousands of languages to romanized text - this model provides a language-universal measure of intelligibility. Utterances with the 5% lowest length-normalized forced alignment score (FAS) are filtered within each language pair.

In Table 5.7, we report the average FAS of the filtered vs accepted portions of XTTS-TEST1 and MMS-TEST, along with Whisper-Large-v3 (Radford et al., 2023) character error rate (CER),

[3]We found anecdotally that the synthesized speech of the embedded language contains elements of native-language influence from the matrix language, as in human accented speech.

Table 5.7: Comparison of the filtered vs accepted synthetic speech. Filtering was done by forced alignment score (FAS).

	XTTS-TEST1				MMS-TEST			
Subset	FAS↑	CER↓	UTMOS↑	SCD↓	FAS↑	CER↓	UTMOS↑	SCD↓
Filtered	-0.68	32.95	2.46	1.12	-0.96	43.10	3.02	1.90
Accepted	-0.26	18.46	2.64	1.07	-0.47	40.07	3.04	1.87

the UTMOS (1-5) (Saeki et al., 2022) automatic naturalness metric, and the number of speakers changes identified by the Pyannote-3.1 (Bredin, 2023) diarization model (SCD). For XTTS-TEST1, filtering has a large effect on Whisper CER and a moderate effect on UTMOS, indicating that the generative synthesis method is prone to producing unintelligible and unnatural speech. On the other hand for MMS-TEST, filtering has less effect on CER and UTMOS, indicating that concatenative synthesis is less error prone; however, SCD shows that this method results in frequent speaker changes.

5.3 Experiments

In this section, we describe several empirical findings based on CS-FLEURS. For ASR, we measure case insensitive and unpunctuated character error rate (CER). For ST (to English), we measure case insensitive and unpunctuated BLEU (Post, 2018).

Benchmarking

How does Whisper perform on code-switched speech as compared to monolingual speech? To answer this, we compare ASR and ST (to English) performance on each CS-FLEURS test set with two monolingual control sets consisting of monolingual speech from all matrix languages in each respective CS-FLEURS set: 1) the original FLEURS and 2) a TTS version of FLEURS, which is monolingual. The latter, referred to as TTS FLEURS, allows us to compare results on TTS-based code-switched speech with the matching TTS-based monolingual speech, eliminating the possibility that Whisper simply is not robust to XTTS or MMS-TTS speech.

As shown in Figure 5.2, ASR CER is over 2x higher on CS-FLEURS than the original monolingual FLEURS - this degradation is largest on the non-English language pair set (XTTS-TEST2). However, the discrepancy between ST performance on monolingual and code-switched speech is low for XTTS and MMS sets. Further, Whisper actually performs better on READ-TEST than the

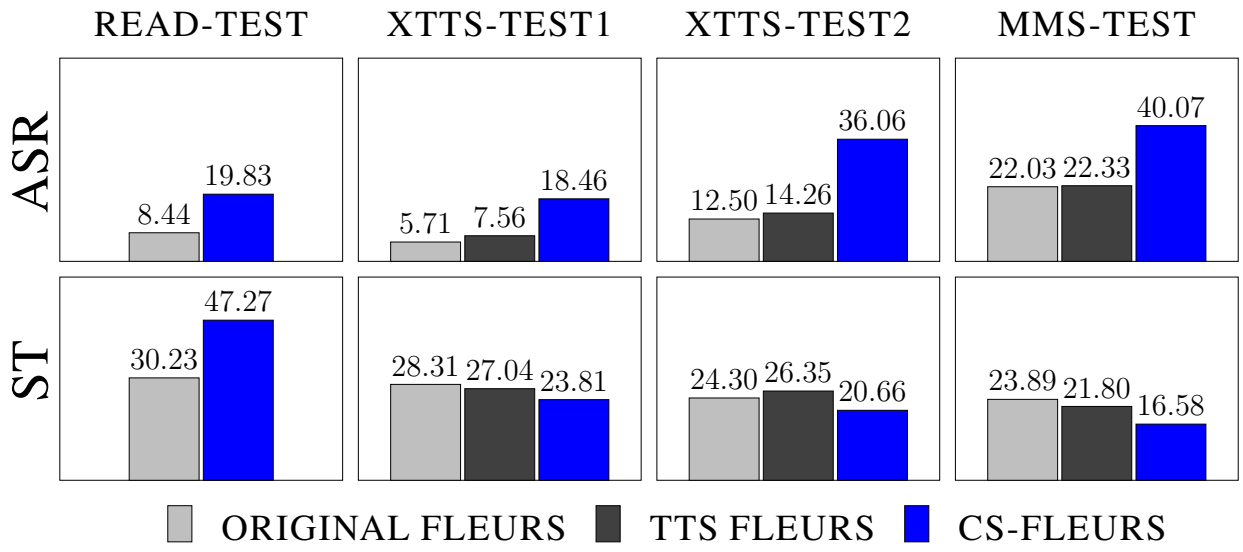

Figure 5.2: Whisper-Large-v3 ASR (CER↓) and ST to English (BLEU↑) on CS-FLEURS test sets (blue). Performance on two monolingual control sets are shown for comparison (gray).

Table 5.8: Comparison of Whisper-Large-v3 ASR (CER↓) performance on same-script vs distinct-script language pairs.

	READ	XTTS		MMS
CS Pair Type	Test	Test1	Test2	Test
Same Script	7.32	8.49	9.92	28.26
Distinct Script	32.33	37.17	40.67	51.62

original FLEURS - this can be attributed to the high rate of X-English code-switching (see Table 5.6). These results indicate that **directly translating code-switched speech is less erroneous than relying on transcription**, although we leave benchmarking ST for non-English targets to future work.

How does Whisper perform on code-switched speech across different language pairs? As shown in Table 5.8, performance on distinct-script language pairs is significantly worse than same-script language pairs - on average CER is 3x higher. These results suggest that **code-switched ASR performance is limited by the ability to interchangeably produce two distinct scripts** within a single utterance-level decoding.

Table 5.9: Results on the effect of training data augmentation.

Data Augment	FLEURS 12 Seen	FLEURS 2 Unseen	CS-FLEURS-READ 12 Seen	CS-FLEURS-READ 2 Unseen
	14.38	8.93	31.77	29.62
✓	**12.67**	**8.51**	**26.24**	**27.77**

Training

Finally, we investigate the question: *how useful is synthetic data for training models?* To answer this, we train two models using the ESPnet toolkit (Watanabe et al., 2018): one using the original FLEURS training data and a second using the original FLEURS plus CS-FLEURS (XTTS-TRAIN). Both models are trained for the same number of iterations following the self-conditioned XLSR-based recipe described in (Chen et al., 2023). We then evaluated on READ-TEST, which consists of 14 language pairs; 12 of these are seen in XTTS-TRAIN and 2 are unseen. We also report the monolingual FLEURS result for the matrix languages of these 12 seen and 2 unseen language pairs. As shown in Table 5.9, **training on synthetic code-switched data improved performance on both seen and unseen language pairs.**

5.4 Conclusion

CS-FLEURS is a hybrid read/synthetic speech dataset for developing and evaluating code-switched systems across 52 languages and 113 unique code-switched language pairs. By using this dataset, which controls for in code-switching patterns, text domain, and audio domain, we find that transcribing speech across distinct-script language pairs remains difficult.

This chapter provides a data setting and baselines for massively multilingual code-switched speech recognition. Next, we discuss proposed work which aims to utilize these resources to enable parallel, temporal multi-sequence modeling for massively multilingual models.

Chapter 6

CS-YODAS: A Mined Dataset of In-the-Wild Code-Switched Speech

Summary

In this chapter we present CS-YODAS, a Creative Commons-licensed dataset of in-the-wild code-switched speech mined from multilingual YouTube data. Code-switching (CS), or the alternation between languages within an utterance or conversation, is common in multilingual settings but remains underrepresented in existing CS speech resources, which are typically small, domain-specific, or artificially constructed. Building on the YODAS corpus, we develop a scalable, human-in-the-loop pipeline for identifying and validating naturally occurring code-switching. The resulting dataset, which totals 313 hours and spans 7 matrix languages, provides diverse, real-world examples of spontaneous code-switched speech. We further analyze the distribution and characteristics of code-switching in the wild, examining language-pair frequencies and switching patterns, and report baseline results for spoken language identification. We hope that CS-YODAS will encourage broader and more comprehensive research on code-switched speech. Dataset link: `https://huggingface.co/datasets/byan/cs-yodas`.

6.1 Introduction

In the current era of large-scale multilingual speech processing, models such as Whisper (Radford et al., 2023), MMS (Pratap et al., 2024), and OWSM (Peng et al., 2025) enable automatic speech recognition (ASR) and language identification (LID) across hundreds of languages. These advances are supported by massive collections of speech, providing broad coverage across languages and domains (Kahn et al., 2020; Pratap et al., 2020b; Ardila et al., 2020; Wang et al., 2021; Li

et al., 2023b).

Yet despite this progress, most large-scale speech resources remain *monolingual* by design: utterances are assigned a single language label, and data mining pipelines are typically optimized to exclude mixed-language content. Consequently, a core feature of multilingual communication, *code-switching*, or the alternation between languages within an utterance or conversation, is largely absent from these corpora.

Collecting code-switched speech at scale presents unique challenges. Unlike monolingual data, it requires bilingual or multilingual speakers who naturally alternate between languages, often in informal or situational contexts. Existing code-switched datasets are therefore limited in scope: they tend to be small, domain-specific, or based on elicited rather than spontaneous speech (Lyu et al., 2010; Deuchar et al., 2014; van der Westhuizen and Niesler, 2018; Chowdhury et al., 2021; Diwan et al., 2021a; Hamed et al., 2022). While synthetic approaches have been proposed to fill this gap, they often fail to capture the spontaneity, prosody, and sociolinguistic nuances of naturally occurring multilingual speech (Hussein et al., 2024b; Yan et al., 2025a).

In this work, we introduce CS-YODAS, a dataset of naturally occurring, in-the-wild code-switched speech mined from multilingual YouTube recordings. Our core motivation is based on the premise that better, more natural code-switched speech data will beget better, more natural code-switched speech systems. We envision that this dataset can be used as a comparison point for synthetically generated code-switching, helping practitioners identify avenues to improve the naturalness of their systems, or as training data for spoken LID systems, improving model robustness beyond what existing domain-specific and synthetically generated data can attain.

Our contributions are summarized as follows:

- A scalable, human-in-the-loop data mining procedure which enables high-precision code-switched detection on in-the-wild data
- The mined code-switched speech dataset, which consists of 313 hours of transcribed speech across 7 matrix languages
- Empirical characterizations of code-switching in the wild, along with baseline evaluations for CS-aware LID systems

Identifying code-switching in large, web-mined corpora presents a major challenge: spontaneous code-switched speech is rare and difficult to detect automatically. Off-the-shelf LID tools struggle with identifying multiple languages within utterances, particularly for noisy in-the-wild data, leading to high false-positive rates. To address this, we design a scalable, human-in-the-loop pipeline that iteratively refines Large Language Model (LLM)-based code-switched detection using human validation. Human feedback is used to guide the LLM via in-context learning, substantially

improving the precision of detected code-switched segments while maintaining scalability across web-scale speech data.

The resulting CS-YODAS dataset reflects a wide range of real-world contexts, including conversational, entertainment, and educational domains, capturing both spontaneous and semi-scripted multilingual interactions. This dataset aims to facilitate future research into both the mechanisms and modeling of natural code-switched speech.

6.2 Dataset Construction

Source Corpus and Motivation

CS-YODAS is derived from the YODAS corpus (Li et al., 2023b), a large-scale multilingual speech resource collected from publicly available YouTube content – by implication, CS-YODAS is under the Creative Commons license. Specifically, we use the cleaned YODAS from the OWSM v4 project (Peng et al., 2025) which consists of 166k hours of audio spanning 75 languages, with speech segments paired to automatic transcripts. While the original corpus organizes data by a single language label, preliminary inspection revealed the presence of code-switching, motivating a systematic effort to identify and extract these examples.

Overview

As shown in Figure 6.1, we apply a LLM-based pipeline to identify code-switched segments: first a text-based LID stage produces a set of candidate segments (*mine_iter0*) and then a human-in-the-loop validation stage produces the final CS-YODAS set (*mine_iter1*). In the remainder of this section, we describe each stage of the pipeline in detail. We also provide example prompts in §6.5.

Mining of Candidate Segments

Each transcript line from YODAS is passed to a multilingual LLM[1] which is prompted to infer the set of languages present in the text. The model outputs both a primary language (the dominant or matrix language) and a list of all other detected languages. Segments where two or more distinct languages are detected are retained as candidate code-switched utterances. This approach leverages the LLM's contextual reasoning and language knowledge to capture switching phenomena that may not be easily identified through token-level LID, such as transliteration, named entities, or embedded phrases.

[1] `https://huggingface.co/Qwen/Qwen3-14B`

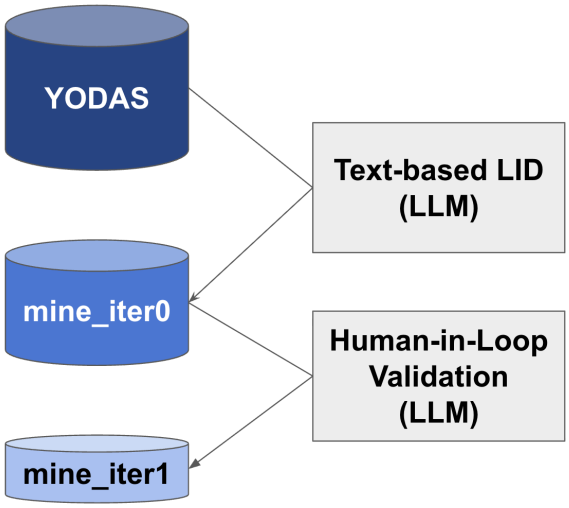

Figure 6.1: Data mining pipeline: we use human feedback on 700 *mine_iter0* utterances as in-context examples while prompting the LLM to validate all of *mine_iter0*, thus producing *mine_iter1*.

However, the in-the-wild nature of the source YODAS data introduces many *distractors*: noisy or imperfect transcripts can falsely mix languages, and borrowed words or named entities often resemble code-switching without representing genuine language alternation. Table 6.1 presents examples of the three most common types: transcription errors, proper nouns, and cognates.

Human-in-the-Loop Validation

To improve precision, we introduce a human-in-the-loop validation procedure. We sample 100 candidate segments across 7 matrix languages for human validation: Arabic, Chinese Mandarin, Czech, French, Hindi, Japanese, Russian. For each segment, annotators are asked 5 questions:

1. Is the transcript correct?
2. Does the segment contain <lang1>?
3. Is <lang1> the matrix language?
4. Does the segment contain <lang2>?
5. Are all <lang2> words proper nouns?

Types	Examples
Transcript Errors	paramことないんですけど今回ね
Proper Nouns	어려운 영어를 참 친엄마 생각하여
Cognates	moment là pour la phase 2 seulement deux

Table 6.1: Examples of common non-CS distractors: words in red caused text-based LID to incorrectly predict the presence of English.

Set	Samples Evaluated	Precision
mine_iter0	700	18.0%
mine_iter1	120	70.0%

Table 6.2: Precision of samples selected from *mine_iter0* vs *mine_iter1*, measured via agreement with human evaluations.

Each of these must be answered with “Yes”, “No”, or “I can’t tell”. Additionally, we solicit comments for each segment where any of Q1-4 were not answered with "Yes".

This human feedback is then utilized via in-context learning when prompting the LLM to generate responses to the same 5 questions for all candidate segments (see §6.5 for example prompts). Note: we also allow the LLM to generate comments. We then filter the candidate segments using the following criteria: Q1-4 = "Yes" and Q5 = "No". The resulting set is taken as the final CS-YODAS set (*mine_iter1*).

Evaluating the Pipeline

We conducted human evaluation on 200 candidate segments drawn from the *mine_iter0* set for each of the seven aforementioned matrix languages. Of these, 100 samples were used as in-context examples for the LLM-based validation round, while the remaining 100 were passed through the full data mining pipeline, with some ultimately appearing in the final validated set, *mine_iter1*.

As shown in Table 6.2, 700 candidate segments from *mine_iter0* were evaluated and only 18% were confirmed to be true instances of code-switching. Following the human-in-the-loop validation, the resulting *mine_iter1* set reached 70% precision across 120 segments identified as code-switched.

Table 6.3 further breaks down the effect of human-in-the-loop validation, providing the full

Lang	TP	FP	FN	TN	Pre. (%)	Rec. (%)
ara	16	13	20	51	55.2	44.4
ces	1	1	1	97	50.0	50.0
cmn	20	6	7	67	76.9	74.1
fra	9	6	13	72	60.0	40.9
hin	34	7	8	51	82.9	81.9
jpn	2	3	10	50	40.0	16.7
rus	2	0	28	70	100.0	6.7
all	84	36	87	458	70.0	49.1

Table 6.3: Full confusion matrix breakdown of the human-in-the-loop stage of the data mining pipeline. TP = True Positive, FP = False Positive, FN = False Negative, Pre. = Precision, Rec. = Recall.

confusion matrix across the 7 matrix languages. Hindi, Chinese, and Arabic yielded the most true positive code-switched samples at precision rates of 82.9%, 76.9%, and 55.2% respectively. Japanese and Russian yielded very few true positives; we found that samples with only proper nouns in the embedded language made up a large portion of the false negatives, indicating some annotator disagreement with our filtering methodology (namely requiring Q5 = "No"). These results highlight variation in the reliability of automatic code-switch detection across matrix languages, underscoring the importance of gathering language-specific human feedback.

Limitations: Recall and Representativeness

While the previous analysis demonstrates substantial gains in precision, evaluating recall in this setting is significantly more challenging. The pipeline operates over a large, weakly supervised corpus of web audio, where the total number of true code-switched segments is unknown. As a result, it is not possible to directly measure how many valid instances of code-switching are missed. The reported recall should therefore be interpreted only with respect to the evaluated subset, rather than as a measure of absolute coverage.

More importantly, recall is shaped by the design of the retrieval pipeline itself. Our approach relies on transcript-based filtering and text-based language identification (LID), which introduces several sources of bias. First, transcripts must be available and sufficiently accurate, which is not guaranteed in many multilingual or informal settings. Second, text-based LID favors segments with clear lexical evidence of multiple languages, and may fail to detect more subtle or phonetically integrated instances of code-switching. Third, alignment quality plays a critical role: short or rapid

Lang	CS Duration	Total Duration	CS Rate (%)
ara	0.6	2.8	21.4
ces	0.0	0.3	13.7
cmn	0.5	4.1	12.1
fra	28.5	273.7	10.4
hin	6.3	21.2	29.7
jpn	0.3	2.0	15.0
rus	1.1	8.9	12.4
all	37.3	312.7	11.9

Table 6.4: Durations (hours) per matrix language: "CS Duration" refers to intra-sentential CS segments while “Total Duration” refers to contextual chunks (CS segment + 15 sec of left/right pad). CS Rate = CS Duration / Total Duration.

switches may be missed if they are not well aligned to the transcript.

These modeling choices implicitly determine what types of code-switching are recoverable. For example, segments containing short insertions, named entities, or phonologically adapted borrowings are more likely to be filtered out, even if they reflect genuine multilingual usage. Conversely, segments with more explicit lexical mixing are more likely to be retained. As a result, the final dataset may not fully reflect the distribution of code-switching observed in the wild.

Taken together, these limitations suggest that CS-YODAS should be viewed as a partial sample of spontaneous code-switched speech. Future work may improve recall and coverage by incorporating speech-based LID, relaxing transcript requirements, or explicitly modeling uncertainty in alignment and segmentation.

Capturing Code-Switching Context

With CS-YODAS, we aim to facilitate the study of when and where code-switching arises in real-world communication contexts. To this end, we construct CS-YODAS in a contextual manner as follows. As described in the previous sections, we first mine a set of target segments with intra-sentential code-switching – these are typically short (<5 sec) utterances. We then extract the surrounding contexts (15 sec before and after) for these target segments and concatenate, producing a contextual chunk. In the case that multiple target segments are within 30 seconds of each other, we simply merge them into one contiguous contextual chunk.

As shown in Table 6.4, the total duration of contextual audio for the seven matrix languages is 312.7 hours. Of this, 37.3 hours (11.9%) correspond to intra-sententially code-switched speech.

Certain languages, such as Arabic and Hindi, exhibit a higher code-switching rate.

Dataset Statistics

Dataset	Matrix Langs	Domain	Speech Type	Hours
ArzEn	1	Conversation	Spontaneous	12
Bangor Miami	1	Conversation	Spontaneous	35
SEAME	1	Conversation	Spontaneous	192
ESCWA	2	Conversation	Spontaneous	3
MUCS	2	Lecture	Structured	160
Soapies	4	TV	Scripted	14
CS-FLEURS	52	Wikipedia	Read/Synthetic	294
- read-set	14	Wikipedia	Read	17
- xtts-set	16	Wikipedia	Synthetic	221
- mms-set	45	Wikipedia	Synthetic	56
CS-YODAS	7	Youtube	Spontaneous	313

Table 6.5: Summary of CS-YODAS vs. CS-FLEURS.

Summary statistics are presented in Table 6.5. The final CS-YODAS release comprises 313 hours of speech across 7 matrix languages.

CS-FLEURS vs CS-YODAS Comparison:

Existing code-switched speech corpora fall along a clear trade-off between linguistic realism and language coverage. Prior spontaneous datasets such as ArzEn, Bangor Miami, and SEAME consist of conversational speech, but are limited to one or two language pairs and relatively narrow domains. Other datasets expand coverage modestly (e.g., Soapies, MUCS), but often rely on scripted or structured speech rather than naturally occurring interactions.

More recent efforts, such as CS-FLEURS (Yan et al., 2025a), significantly increase language coverage by constructing code-switched data at scale. CS-FLEURS spans dozens of languages (up to 52 matrix languages) and hundreds of hours by generating code-switched text from Wikipedia and realizing it through read speech and text-to-speech synthesis. This enables controlled evaluation across many language pairs, but the resulting speech reflects the constraints of its construction: it is read or synthetic, tied to a written domain, and lacks the spontaneity of real conversational code-switching.

In contrast, CS-YODAS focuses on capturing code-switching as it occurs in practice. It consists of over 300 hours of naturally occurring speech from YouTube across multiple domains, including informal conversation, commentary, and mixed-media content. While its language coverage (7 matrix languages) is smaller than CS-FLEURS, it provides spontaneous, in-the-wild examples of when and how speakers switch languages, including phenomena that are difficult to simulate through text generation alone.

Taken together, these datasets occupy complementary positions. CS-FLEURS enables large-scale, controlled evaluation across many languages, while CS-YODAS captures the variability and unpredictability of real-world code-switching. This distinction is critical: models that perform well on synthetic or read code-switching do not necessarily generalize to spontaneous settings, where language alternation is driven by discourse, context, and speaker intent rather than predefined text.

6.3 Dataset Analyses

Overview

The primary motivation of our dataset analysis is to augment the dataset with additional metadata, such as language roles, parts-of-speech, and text or audio domains. In this section, we describe our methodology for generating metadata and summarize the metadata distributions.

Please be aware that the following analyses are based on data which we estimate to be of 70% precision (as described in section §6.2). Please also refer to §6.5 for further discussion of the limitations of this dataset.

Language Distributions

First, let's address the obvious question: how often did code-switching occur in-the-wild? Given that our method prioritizes precision over recall we cannot directly answer this question. Instead, we try to provide a lower-bound estimate by reporting the data mining yield rate (i.e. the number of segments identified as code-switched out of the total number of source segments) – we expect that the actual rate of code-switching in-the-wild is somewhat higher than the yield rate. As shown in Figure 6.2, the yield varied greatly across matrix languages, ranging from 0.01% for Russian to 6.87% Hindi. Arabic (1.64%), French (0.66%), and Chinese (0.52%) also exhibited moderate yield rates.

Next, let's examine the distribution of embedded languages. English is by far the most common embedded language, accounting for 85.6% of the data. Figure 6.3 shows the English vs. Non-English embedded language breakdown across the 7 matrix languages. Arabic (27.6%) and

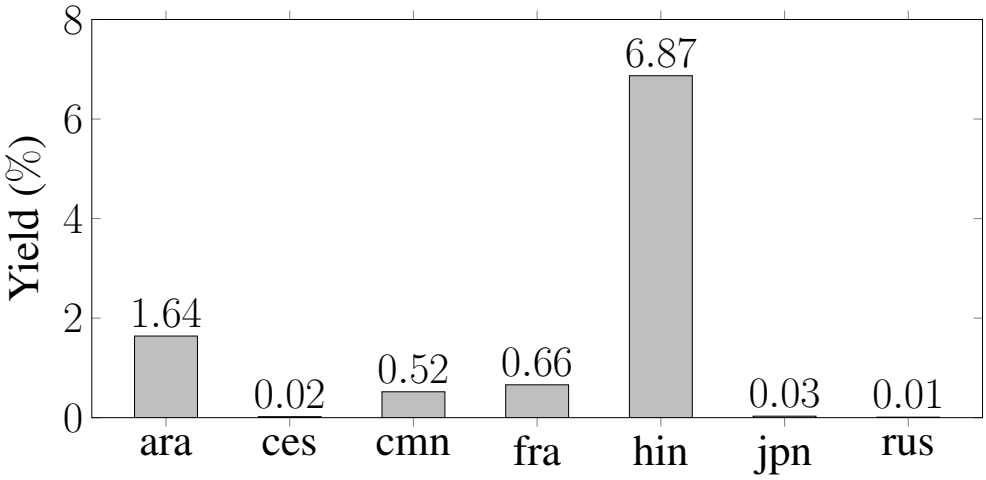

Figure 6.2: Yield (%) of segments identified as code-switched out of the total number of segments across matrix languages.

French (14.6%) exhibit the highest rates of code-switching with non-English languages. For Arabic, Egyptian Arabic accounted for the highest proportion of non-English code-switching. For French, Arabic accounted for the highest rate of non-English code-switching.

Lexical Distributions

Next, we examine the lexical distribution of the embedded English words to better understand the conversational purpose of code-switching. Our analysis focuses on English words to ensure reliable part-of-speech (POS) tagging. We first extract all English words and run POS tagging using spaCy. Each word is then assigned to one of four categories: *content words*, *function words*, *proper nouns*, and *discourse markers*. Content words include nouns, verbs, adjectives, and adverbs. Proper nouns are treated as a separate category. Function words include determiners, adpositions, conjunctions, particles, and auxiliaries. Finally, discourse markers include interjections such as "yeah", "like", "well", etc.

Figure 6.4 plots the proportions of the four lexical categories for both CS-YODAS and CS-FLEURS; the denominator here is the total count of embedded English words. Both sets contain mostly content words and proper nouns among their embedded English words. However, CS-YODAS shows a much smaller proportion of function words, nearly half that of CS-FLEURS – this suggests that the lexical pattern of the synthesized CS-FLEURS text may be unnatural. This finding aligns with linguistic theory, which suggests that function words should be least commonly used in the embedded language (Myers-Scotton, 1993). Finally, CS-YODAS includes some discourse

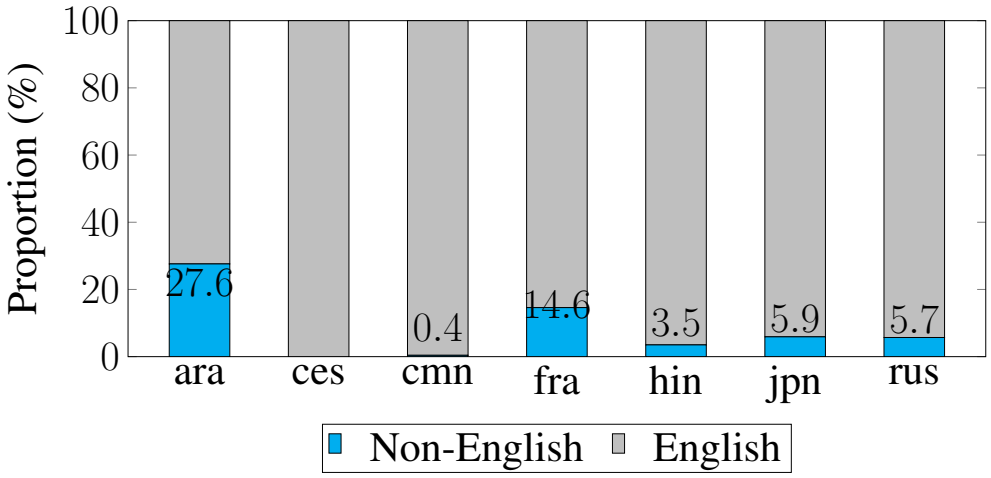

Figure 6.3: Proportions (%) of examples with English vs. Non-English as the embedded language across matrix languages.

markers, which are typical in conversational speech, whereas CS-FLEURS has none, consistent with its synthetic nature.

Domain Distributions

To further characterize the dataset, we obtain topical domain labels by feeding transcripts to a multilingual text-based domain classification model.[2] We then use these domain labels to estimate where code-switching tends to occur. To do so, we compare the proportional representation of each domain in CS-YODAS to that in the original YODAS corpus. For each domain, we compute the ratio between its relative frequency in CS-YODAS and its relative frequency in YODAS. A value greater than one indicates that the domain is over-represented in CS-YODAS relative to the source YODAS data, while values below one indicate under-representation. This approach normalizes for overall corpus composition and highlights domains where code-switching occurs disproportionately often.

As shown in Figure 6.5, informal and tech-oriented domains such as *Games*, *Internet and Telecom*, and *Computers and Electronics* show the strongest over-representation, suggesting that code-switching frequently arises in conversational, online, and technology-related contexts. Moderately elevated ratios are also observed in *Sports*, *Business*, and *News*, where English terms or expressions are often borrowed. In contrast, more formal or topic-specific domains, such as *Law*

[2] `https://huggingface.co/nvidia/multilingual-domain-classifier`

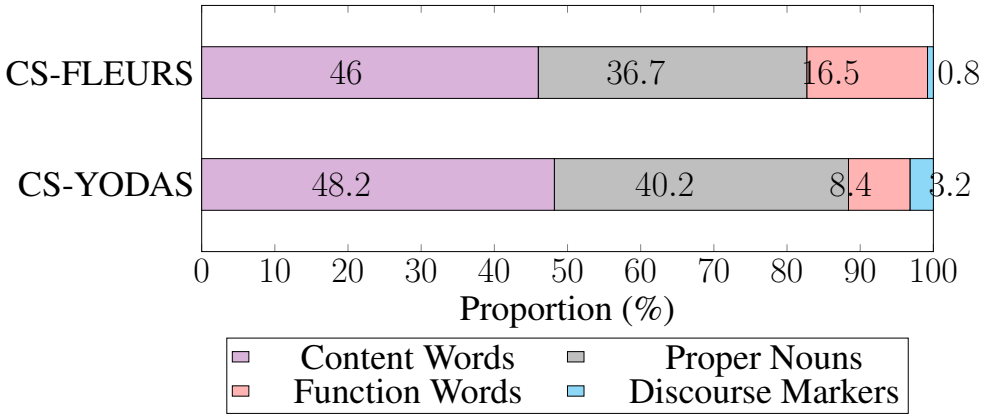

Figure 6.4: Proportions (%) of embedded English words across 4 categories: content words, proper nouns, function words, and discourse markers.

and Government, *Health*, *Science*, and *People and Society*, show lower ratios, indicating that code-switching is less common in structured or institutional settings.

6.4 Baseline Experiments

Train Set	FLEURS	CS-FLEURS							
		XTTS1	MMS	READ					
				ara-eng	cmn-eng	fra-eng	hin-eng	jpn-eng	rus-eng
w/o CS-YODAS	97.4	99.7	99.9	0	0	0	0	0	0
w/ CS-YODAS	96.3	99.5	99.8	0	0.3	51.1	19.3	0	0

Table 6.6: Comparison of spoken LID models with and without CS-YODAS training data. Accuracy (%) is reported on FLEURS and CS-FLEURS.

Overview

Modern multilingual speech recognition and translation systems rely on accurate spoken LID both for curating large-scale training datasets (Valk and Alumäe, 2021; Peng et al., 2025) and for routing input to the language-specific modules (Radford et al., 2023; Pratap et al., 2024). However, current spoken LID systems (Jia et al., 2023; Wang et al., 2025) are almost exclusively trained under the

assumption that utterances are purely monolingual, ignoring the possibility that a single utterance may contain multiple languages. Prior work has shown that this monolingual bias greatly limits the quality of downstream speech processing for code-switched speech (Peng et al., 2023; Yan et al., 2025a).

This limitation reflects a broader gap in LID research. Burchell et al. (2024) show that even in the text domain, utterance-level code-switched LID remains challenging. In the spoken domain, prior LID studies (Rangan et al., 2020; Li et al., 2023a) are limited to only a few language pairs. We argue that this gap primarily stems from a data bottleneck: large-scale multilingual speech corpora with natural code-switching and reliable language labels have been lacking.

In this work, we leverage CS-YODAS alongside the monolingual FLEURS dataset and the synthetic code-switched CS-FLEURS dataset to establish baseline LID systems that explicitly account for both monolingual and code-switched utterances. Our findings show that synthetic code-switched training data alone is insufficient to support robust generalization, and that exposure to spontaneous, naturally occurring code-switching is essential. These results highlight the importance and feasibility of curating large-scale spontaneous code-switched resources to enable more realistic spoken LID in the wild.

Experimental Setup

We compare two training data configurations: (1) simply combining the FLEURS (Conneau et al., 2023) training set with the XTTS-generated training data from CS-FLEURS and (2) further including CS-YODAS data.[3] Both configurations contain 102 monolingual languages and 16 code-switched language pairs. The 16 pairs originate from CS-FLEURS, with 6 of them further supplemented by data from CS-YODAS. We refer to these settings as "w/o CS-YODAS" and "w/ CS-YODAS" respectively.

Our LID models are based on self-supervised representations, with MMS (Pratap et al., 2024) as the upstream encoder and ECAPA-TDNN (Desplanques et al., 2020) as the downstream embedding extractor. Classification is performed using the AAMSoftmax loss (Deng et al., 2019) with the sub-center enhancement (Zhao et al., 2021), following the ESPnet-SPK implementation (Jung et al., 2024). For code-switched utterances, each language pair is treated as a distinct class (i.e. "cmn" and "cmn-eng" are distinct). Training hyperparameters and optimization settings are kept consistent with the VoxLingua107-only setup described in Wang et al. (2025). The final models are selected based on the best accuracy on the FLEURS dev set and CS-YODAS XTTS1 test set. Models are evaluated on FLEURS and CS-FLEURS test sets.

[3]We only use the examples with English as the embedded language for this experiment, and exclude ces due to its limited data.

Results and Discussion

As shown in Table 6.6, the model "w/o CS-YODAS" performs well on in-domain settings, including FLEURS and the CS-FLEURS XTTS1, as well as the out-of-domain synthetic CS-FLEURS MMS. However, the model fails across all languages in CS-FLEURS READ; the model is unable to distinguish code-switched utterances. We posit that this is due to a domain mismatch resulting from a lack of natural code-switched training data.

After incorporating the CS-YODAS training data, performance improves on the fra–eng (0% to 51.1%) and hin–eng (0% to 19.9%) subsets of the CS-FLEURS READ. This *zero-to-one* improvement demonstrates that exposure to in-the-wild code-switched speech enables models to begin generalizing to natural switching patterns beyond what synthetic data can support.

As shown in Figure 6.6, there is a clear trend between code-switched training data duration and code-switched LID performance: accuracy does not rise past 0 until after 5 hours of code-switched training data is available.

Our hope is that these baselines demonstrating the impact of in-the-wild code-switched training data motivates future dataset construction efforts, as our belief is that solving the data problem is critical towards building effective code-switched spoken LID systems.

6.5 Conclusion

In summary, our work introduces CS-YODAS, a large-scale dataset of spontaneous, naturally occurring code-switched speech. Using CS-YODAS along with the existing CS-FLEURS, we are able to show that synthetically generated code-switching has yet to thoroughly mimic the patterns observed in the wild. Further, we show that synthetic code-switched training data has yet to obviate the need for collecting spontaneous code-switched speech. Our hope is that CS-YODAS facilitates future research towards building robust spoken LID systems capable of handling all nuances of multilingual communication.

Limitations

While CS-YODAS represents a significant step toward large-scale, naturally occurring code-switched speech resources, several limitations remain.

First, the underlying source corpus, YODAS, is based on Creative Commons YouTube content, which inherently biases the dataset toward publicly available and broadcast-style material, such as news, educational talks, and announcements. While the dataset still captures code-switching in daily, informal contexts, we suspect that the casual setting may be underrepresented.

Second, while we implemented a human-in-the-loop validation pipeline to improve the precision of mined code-switched examples, the scale of the data precludes exhaustive manual verification. We also acknowledge other the presence of residual noise from the source corpus, such as transcription and audio segmentation errors.

Finally, reliable evaluation of code-switching requires skilled bilingual annotators. This reliance on language-specific expertise is a limiting factor against scaling coverage, and thus this release of CS-YODAS spans only 7 matrix languages.

Ethics Statement

This dataset is a mined subset from an already publicly released dataset, so we do not foresee any harm arising from its content. Annotators were recruited for this project on a volunteer basis and were made aware of our data mining approach and thus agreed to the use of their feedback to refine the dataset.

Example LLM Prompts

Overview

This section provides examples of the prompts used for text-based LID and human-in-the-loop Validation, supplementing the main description of the LLM-based data mining pipeline in §6.2.

Text-based LID

Table 6.7 illustrates an example prompt used for the first stage of our data mining pipeline. We task the LLM with identifying all languages that appear in the segment. We use only 3 in-context examples in this stage; this choice was made in an attempt to maximize throughput.

Human-in-the-Loop Validation

Table 6.8 illustrates an example prompt used for the second stage of our data mining pipeline. We task the LLM with replicating the human feedback pattern. We use 100 in-context examples in this stage, sourced from the human feedback on same-language samples; this choice was made in an attempt to maximize accuracy.

(System) You are performing text-based language identification. We are trying to identify code-mixed or code-switched utterances.

(User) Text: “‘这些 地区 人口 稀少， often 不 存在 光 污染 的 问题， 你 also 能 欣赏 到 璀 璨 星空 。“‘

For the given text in triple backticks identify ALL languages that appear. There may be only a single language or multiple languages that are code-mixed together. Your final answer should list the languages in order of prevalence. Code-mixing, or code-switching, is defined as the alternation of two languages within a single discourse, sentence, or constituent. Double check whether the text contains code-switching by reviewing word-by-word. Do not simply glance at the overall sentence and only write down the dominant language.
Format your response as a json object.

(Assistant) {"languages": ["Chinese", "English"]}

... 3 in-context examples are provided in total ...

(User) <Prompt with target segment>
(Assistant) <Text-based LID output>

Table 6.7: Example prompt used for text-based LID. In-context examples are depicted in blue and target segment prompt/response are depicted in violet.

(System) You are performing data validation. We are looking for code-switched speech examples by examining transcript text.

(User) Text: “‘因此受到他的祝福之下就代表的是权威Authority“‘

For the given text in triple backticks, answer the following 5 questions:
1. Is the transcript correct?
2. Does the speech contain Chinese?
3. Is Chinese the matrix language?
4. Does the speech contain English?
5. Are all English words proper nouns?

(Assistant) {"Q1": "Yes", "Q2": "Yes", "Q3": "Yes", "Q4": "Yes", "Q5": "No", "Comments": ""}

... 100 in-context examples are provided in total ...

(User) <Prompt with target segment>
(Assistant) <Human-in-the-loop Validation output>

Table 6.8: Example prompt used for human-in-the-loop validation. In-context examples are depicted in blue and target segment prompt/response are depicted in violet.

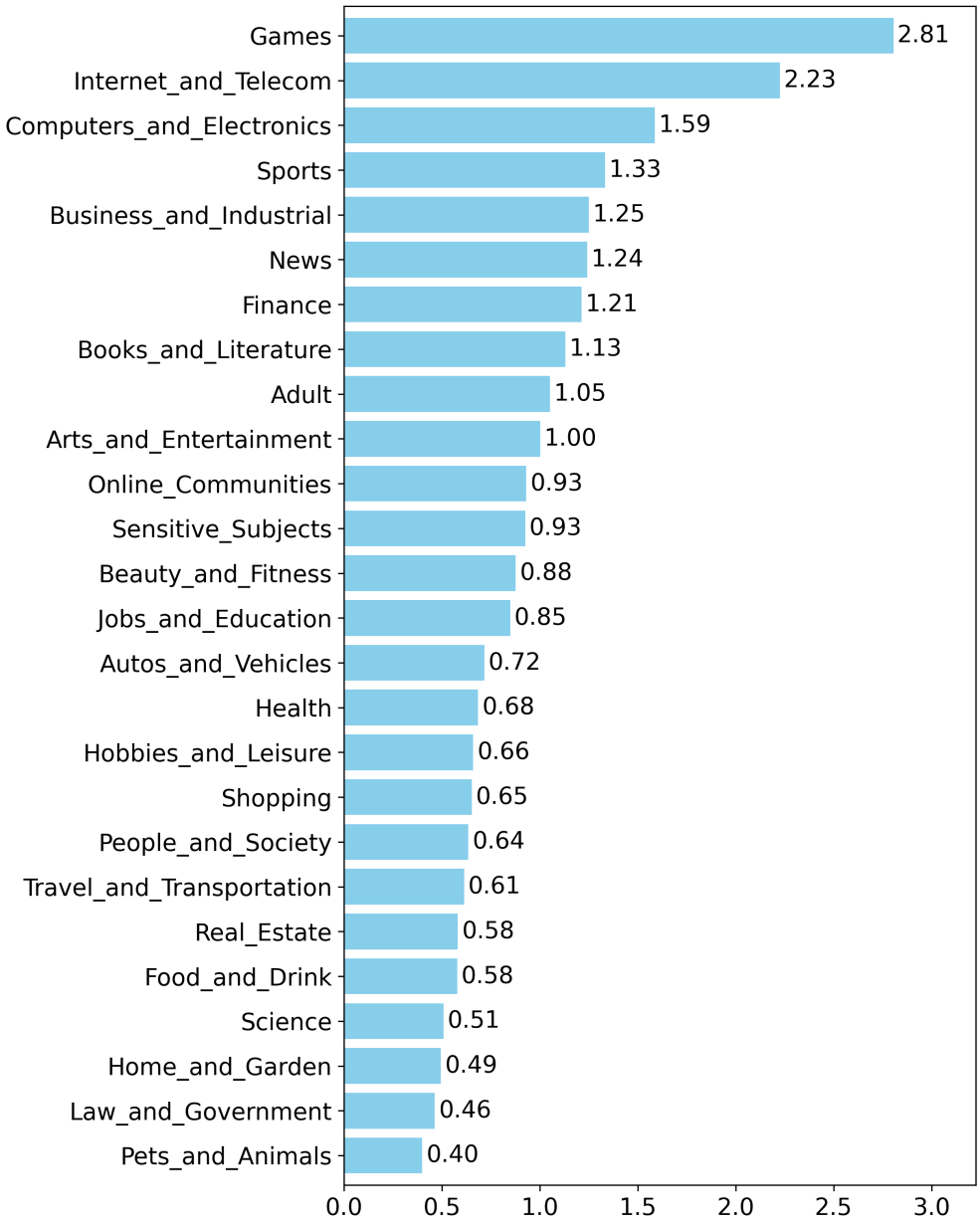

Figure 6.5: CS likelihood by domain (CS-YODAS proportion / YODAS proportion): values greater/less than 1 suggest higher/lower CS likelihood.

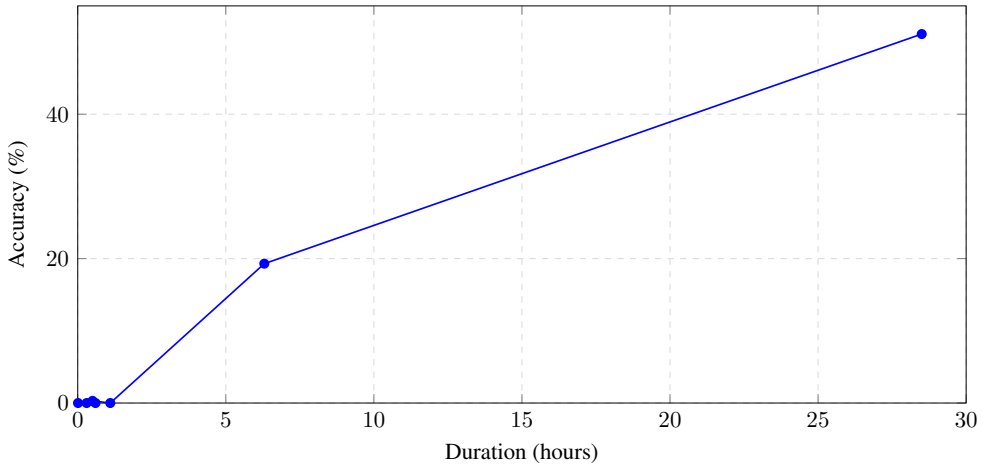

Figure 6.6: Accuracy (%) on CS-FLEURS READ vs. Duration (hours) of CS-YODAS training data.

Chapter 7

CS-Anything: Towards Massively Scaling Language-Factorized ASR

Summary

Massively multilingual speech models support over 100 languages, yet still underperform on code-switched speech. We argue that this gap is structural rather than purely data-driven: systems such as Whisper select a single language mode for the entire utterance, presuming that language identity never changes within a sequence. We present a scalable language-factorized ASR framework that combines frame-level Language Diarization with Language-Diarization-Conditioned Whisper (LaDiCoW). Experiments on 18 languages across CS-FLEURS and additional conversational benchmarks show that modeling language identity over time significantly reduces the performance gap between monolingual and code-switched speech.

7.1 Introduction

In the current era of massively multilingual speech foundation models (Li et al., 2022b; Radford et al., 2023; Omnilingual et al., 2025) that operate on 100+ languages, is code-switched speech recognition already solved? On the surface, it seems that any entity (human or machine) that has mastered *N* languages would also be capable of understanding any of the $\binom{N}{2}$ code-switching combinations. Yet recent investigations (Yan et al., 2025a) indicate that state-of-the-art transcription models such as Whisper (Radford et al., 2023) still perform significantly worse on code-switched speech compared to monolingual speech with identical topical domains and audio conditions. This performance discrepancy motivates the core question behind this work: **what is prohibiting modern transcription systems from mastering code-switching?**

A simplistic answer would be data scarcity: a recent study (Yan et al., 2026) estimated that less than 2% of publicly available YouTube data contained code-switching. One may argue that since code-switching is rare and only found in particular bilingual conversational settings, it should not be surprising that models trained on web-scale data under-perform on this corner case of speech – in this work we offer an alternative to this status quo.

Our view is that modern transcription systems under-perform for code-switching because they lack language diarization, or the ability to determine what language is being spoken *and when*. Therefore, in this work we seek to demonstrate that **language diarization is the critical component needed to unlock massively multilingual code-switched transcription.**

To illustrate why language diarization is critical, we need to first highlight the fundamental incompatibility of modern multilingual system design with code-switched speech. Systems like Whisper rely on language-specific transcription modes, commonly implemented via language ID tokens which auto-regressively condition transcript generation, which are statically set at an utterance-level (as opposed to a more fine-grained frame-level). Workarounds like setting multiple utterance-level language IDs at once can improve performance (Peng et al., 2023), but not to the degree where transcriptions are reliable.

Alternatively, if a system has language diarization then it can apply the appropriate language-specific transcription modes on the respective portions of the code-switched utterance. These intermediate monolingual transcriptions are then merged into the ultimate code-switched transcription. This approach, coined as *language-factorized ASR*, was first demonstrated on the bilingual setting (Yan et al., 2022b). In fact, it was shown that such systems can be constructed with minimal code-switched data (Yan et al., 2023c), refuting the aforementioned view that code-switched ASR is entirely bottle-necked by data scarcity. The key intuition of the approach, which is to break the bilingual task into monolingual parts, was also validated in related prior works (Lu et al., 2020; Zhou et al., 2020b; Dalmia et al., 2021a; Tian et al., 2022; Song et al., 2022), and several works already shown that bilingual decomposition can be done using language diarization (Liu et al., 2021, 2023, 2025).

In this work we make the following contributions:

- Formulation of large-scale language-factorized ASR, leveraging the 100+ language Whisper foundation model
- Empirical validation on the broadest existing code-switched ASR benchmark, CS-FLEURS (Yan et al., 2025a), along with additional bilingual conversational sets, for a total of 18 languages[1]

[1]There is potential for supporting 100+ languages, but this work was limited by the coverage of synthetic code-switched training data.

- Release of open-source models, data, and infrastructure

Our hope is that these efforts encourage further developments of flexible multilingual speech processing models which are robust to any and all types of code-switched speech (Auer, 2013).

7.2 Background and Motivation

Language-Factorized Bilingual ASR

We first review the conditionally factorized framework for bilingual ASR proposed in (Yan et al., 2022b). Let X denote speech features and $Y = \{y_n \in (\mathcal{V}^M \cup \mathcal{V}^E) | n = 1, ..., L\}$ denote bilingual transcriptions, where $\mathcal{V}^M$ and $\mathcal{V}^E$ are the Mandarin and English vocabularies, respectively. The transcription Y may be purely monolingual or code-switched.

Rather than directly modeling $p(Y|X)$, the key idea is to introduce monolingual label-to-frame alignment sequences $Z^M = \{z_t^M \in \mathcal{V}^M \cup \{\varnothing\} | t = 1, ..., T\}$ and $Z^E = \{z_t^E \in \mathcal{V}^E \cup \{\varnothing\} | t = 1, ..., T\}$, where $\varnothing$ denotes the CTC blank symbol. Under the assumption that Z^M and Z^E are conditionally independent given X, Viterbi approximation yields:

$$\hat{Z}^\ell = \arg\max_{Z^\ell} \sum_{t=1}^{T} \log p(z_t^\ell | X), \quad \ell \in \{M, E\}. \tag{7.1}$$

The final transcription Y is then obtained by deterministically merging $\hat{Z}^M$ and $\hat{Z}^E$. Given these monolingual alignments, no additional information from X is required to determine the output Y, a property we refer to as *language-factorized*.

Limitations of the Bilingual Formulation

While the bilingual conditional factorization is elegant, it faces fundamental scalability limitations. The framework requires one monolingual encoder per language, meaning that supporting N languages would require N separate encoders in addition to a bilingual decoder for each of the $\binom{N}{2}$ possible pairs.

Moreover, the original formulation relies on CTC-based label-to-frame synchronization to provide ordering information, which constrains the choice of ASR architecture to CTC models. Modern large-scale ASR foundation models such as Whisper (Radford et al., 2023) use an encoder-decoder attention architecture which fundamentally differs from the non-autoregressive CTC models.

These limitations motivate our reformulation, which replaces CTC-based bilingual synchronization with explicit language diarization and leverages Whisper as the ASR backbone.

7.3 Proposed Framework

Overview

Our goal is to preserve the principle of language factorization while removing the aformentioned scalability bottlenecks (§7.2). Instead of implicitly separating languages through monolingual encoders and CTC alignment, we make frame-level language identity an explicit, standalone classification problem. This leads to a two-stage decomposition that scales more naturally to 100+ languages.

Stage 1: Language Diarization (LaDi)

We define language diarization as frame-level language ID. Let $\mathcal{L}$ denote the set of known languages. Given speech features $X = \{\mathbf{x}_t \in \mathbb{R}^D \mid t = 1, \ldots, T\}$, the language diarization output is $D = \{d_t \in \mathcal{L} \cup \{\text{silence, noise}\} \mid t = 1, \ldots, T\}$, where each frame is assigned a language label or a non-speech label. The language diarization network is initialized from the Whisper encoder, which produces 50 frames per second, and uses a frame-level projection head to produce D via argmax.

The diarization output D is then segmented into per-language time intervals. For a language $\ell \in \mathcal{L}_{\text{active}}$, we derive language-specific segments $D^\ell = [(s_1, e_1), \ldots, (s_K, e_K)]$ consisting of K time intervals (start, end) where language ℓ is active. These segments serve as the language diarization conditioning signal for the downstream ASR model.

Stage 2: LaDi-Conditioned Whisper (LaDiCoW)

We adapt the DiCoW (Diarization-Conditioned Whisper) architecture (Polok et al., 2026), originally designed for speaker-attributed ASR, to our language-factorized ASR setting. In the original DiCoW, frame-level speaker diarization masks indicating Silence-Target-Non-target-Overlap (STNO) categories are injected into each Whisper encoder layer via frame-level diarization-dependent transformations (FDDT). FDDT applies category-specific affine transformations weighted by the diarization probabilities at each frame. We re-purpose this mechanism for language conditioning. For a given target language ℓ, we construct a language diarization mask analogous to the speaker STNO mask, where frames are categorized as: silence (no speech), target language ℓ, or non-target language (any other language).[2] The FDDT layers then adapt the encoder representations to focus on the target language, enabling target-language ASR.

[2]Unlike in speaker diarization, we do not expect any overlaps in our formulation of language diarization.

For each language detected in the diarization output, $\ell \in \mathcal{L}_{\text{active}}$, LaDiCoW performs target-language ASR producing segment-level alignment sequences $Z^{\ell}_{\text{seg}} = \{z^{\ell}_{\text{seg},k} \in \mathcal{V}^{\ell} \cup \{\text{time-stamps}\} \mid k = 1, ..., K\}$, where each $z^{\ell}_{\text{seg},k}$ represents a segment of transcription and with predicted time-stamps. The sequence is obtained via auto-regressive decoding:

$$\hat{Z}^{\ell}_{\text{seg}} = \arg\max_{Z^{\ell}_{\text{seg}}} \sum_{k=1}^{K} \log p(z^{\ell}_{\text{seg},k} \mid z^{\ell}_{\text{seg},<k}, X, D^{\ell}). \tag{7.2}$$

This new formulation in equation 7.2 differs from equation 7.1 in two critical ways:

- Language diarization outputs, D^{ℓ}, are now additionally provided as a conditional variable.
- Segment-level alignment sequences for each language, Z^{ℓ}_{seg}, are now produced auto-regressively.

In other words, we've reformulated the task of generating monolingual target-language alignment sequences to be compatible with the LaDiCoW backbone. The final code-switched transcription Y is then composed by interleaving the sets of segment-to-frame alignments, Z^{ℓ}_{seg}, across all active languages, $\mathcal{L}_{\text{active}}$, according to the chronological order of the predicted time-stamps for each segment.

7.4 Experimental Setup

Datasets

The same data is used for Stage 1: LaDi and Stage 2: LaDiCoW. Experiments are based on a diverse set of code-switched benchmarks which are summarized as follows.

In total, there are 7 different test sets (14 languages):

- **CSFL-READ** (Yan et al., 2025a): consists of 12 matrix languages[3] with English embedded. This is the broadest single set of code-switched speech across languages, collected via read speech on generated Wiki-domain text; we use this set to compare performance across languages without interference from topical domain or audio condition differences.
- **SEAME** (Lyu et al., 2010): consists of the **SGE** set which is matrix (Singaporean) English with embedded Mandarin and the **MAN** set which is matrix Mandarin with embedded English. The speech is conversational.
- **ARZN** (Hamed et al., 2022): matrix Egyptian Arabic with embedded English. The speech is conversational.

[3]Arabic, Czech, Mandarin, German, French, Hindi, Italian, Japanese, Korean, Portuguese, Russian, Spanish

- **MUCS** (Diwan et al., 2021a): consists of the **HIN** set which is matrix Hindi with embedded English and the **BEN** set which is matrix Bengali with embedded English. The embedded English found in these lecture-style sets are often technical or otherwise rare terms.

We consider two training data settings:

- **CSFL** (17 langs): training only on the CSFL synthetically generated data. Note that the CSFL training data covers all the languages in the conversational sets except for Bengali.
- **All** (18 langs): training on CSFL, SEAME, ARZN, MUCS.

These two data settings allow for the comparison of in-domain versus out-domain performance on the SEAME, ARZN and MUCS sets; furthermore, it can demonstrate the efficacy of synthetic-only training data towards general performance.

Data Preparation

All of our code-switched corpora were created originally for code-switched ASR: the corpora provide speech segments with their corresponding transcripts. To support the training and evaluation of LaDi and LaDiCoW systems, we first constructed **silver** language diarization labels using the following strategy:

1. We first align speech to tokenized transcripts using the MMS universal forced alignment model (Pratap et al., 2024). This CTC-based alignment model operates on Romanized text and was trained on thousands of languages. We therefore consider this to be a multi-lingually robust choice for our code-switched data.
2. We then run text-based language ID using the FastText (Joulin et al., 2017) model on individual tokens, restricting the classification to the expected languages for each set.

Combining token-to-speech alignment (step 1) with token-level language ID (step 2) yields language diarization labels - we label these as "silver" rather than "gold" since they were not produced directly via human annotation.

Metrics

For language diarization, we report **Jaccard Error Rate (JER↓)** (Ryant et al., 2019; Watanabe et al., 2020). Let R_ℓ and P_ℓ denote the reference and predicted time regions for language ℓ. JER is defined as one minus the mean intersection-over-union across all L languages:

$$\text{JER} = 1 - \frac{1}{|L|} \sum_{\ell \in L} \frac{|R_\ell \cap P_\ell|}{|R_\ell \cup P_\ell|} \tag{7.3}$$

Unlike diarization error rate, which is dominated by majority classes, JER's intersection-over-union formulation balances performance across languages regardless of how much they are spoken. This is particularly important in code-switched speech, where embedded languages often occupy substantially less duration than the matrix language.

For ASR, we report **Character Error Rate (CER↓)**.[4] We select CER rather than Word Error Rate because it provides a consistent unit of measurement across languages with different writing systems and tokenization conventions. CER enables direct comparison and aggregation across languages without relying on language-specific tokenization rules.

Comparisons

We compare our LaDi systems to two baselines:

- **ECAPA-TDNN** (Ravanelli et al., 2021): a spoken language ID system run with two temporal resolutions (0.25s and 8s windows).
- **Whisper Pipeline**: a baseline which first runs default Whisper ASR and then derives language diarization predictions via forced alignment and token-level language ID (using a similar process as described for silver labels §7.4).

These baselines are limited: (i) neither are not trained on code-switched data and (ii) ECAPA-TDNN is an utterance-level language ID model. We therefore also provide two hypothetical systems (referred to as strawmen) for comparison:

- **Matrix Language**: predicts the matrix ℓ for all frames.
- **Embed Language**: predicts the embedded ℓ for all frames.

The Matrix Language strawman is a proxy for the best that an utterance-level spoken language ID system could do on the code-switched language diarization task.

For ASR, we compare with four baselines:

- **Whisper (Seg)**: a baseline which first uses silver language diarization to segment utterances and then runs Whisper ASR on those monolingual segments individually.
- **Whisper (Utt)**: a baseline which runs Whisper ASR on code-switched utterances. We force Whisper to use the matrix language mode rather than relying on Whisper's language ID predictions which can be unreliable (Yan et al., 2025b).

[4]Evaluation is performed after removing punctuation, casing, and white-space to avoid penalizing formatting differences.

- **Whisper-FT (CSFL)**: Whisper fine-tuned on CSFL only.
- **Whisper-FT (All)**: Whisper fine-tuned on all data.

The two fine-tuned Whisper models are the main comparison points since they utilize the same training data, but the other two baselines are nonetheless useful points of reference.

7.5 Results and Analyses

We first present results on language diarization and ASR individually, where diarization-conditioned ASR systems utilize only silver diarizations. We then present results on the cascade, where diarization-conditioned ASR systems utilize predicted diarizations. Breakdowns by language are also presented.

Language Diarization

Table 7.1: Language Diarization (JER↓) Performance. Lighter Cells = Out-Domain. Darker Cells = In-Domain.

Type	Model	CSFL	SEAME		ARZN	MUCS	
		READ	SGE	MAN	ARA	HIN	BEN
Straw	Matrix Language	67.45	43.29	59.00	47.73	53.27	59.29
Straw	Embed Language	82.40	80.26	75.42	86.69	86.23	84.28
Base	EcapaTDNN (8s)	65.32	91.29	81.90	69.28	70.03	58.17
Base	EcapaTDNN (0.25s)	99.00	98.28	97.51	99.05	99.66	99.14
Base	Whisper Pipeline	41.52	31.18	34.25	49.59	44.23	79.35
Ours	LaDi (CSFL)	27.59	61.61	48.08	47.49	56.07	96.26
Ours	LaDi (All)	19.61	16.47	16.19	22.05	28.25	21.77

In Table 7.1, we compare LaDi to several strawmen and baseline systems (described in §7.4). As expected, while neither strawman performs well, predicting the matrix language results in a lower JER than assigned to the embedded language. The baselines using the EcapaTDNN utterance-level spoken language ID system also perform poorly: choosing a wide window of 8s

mimics the matrix language strawman to a certain degree, albeit with additional language confusions, while choosing a narrow window of 0.25s results in a highly noisy diarization which suggests that the system requires more context to make reliable classifications. The strongest baseline is the Whisper Pipeline, which can successfully diarize portions of code-switched speech but only if Whisper produced correct code-switched transcriptions, which is not guaranteed.

The performance of the LaDi systems exhibit a large effect of in-domain training. For instance, LaDi (CSFL) is significantly worse on the SEAME-SGE set compared to LaDi (All): this can be partly attributed to the mismatch between the Mandarin-English data seen during training (matrix Mandarin) and testing (matrix English). Other factors such as audio conditions and speaker differences (particularly accents) likely also contribute to the lack of out-domain generalization shown by LaDi (CSFL). On the other hand, the LaDi (All) system exhibits relatively low JER across all sets.

ASR

Table 7.2: ASR (CER↓) Performance. Lighter Cells = Out-Domain. Darker Cells = In-Domain. Language Diarization (LD) column indicates use of (S)ilver or (P)redicated.

Type	Model	LD	CSFL	SEAME		ARZN	MUCS	
			READ	SGE	MAN	ARA	HIN	BEN
Base	Whisper (Seg)	S	23.84	37.44	43.45	39.69	75.10	167.3
Base	Whisper (Utt)	-	14.25	63.76	74.46	90.91	53.17	91.43
Base	Whisper-FT (CSFL)	-	6.91	36.86	31.31	28.87	56.70	103.18
Base	Whisper-FT (All)	-	6.68	12.27	10.88	14.06	30.98	20.80
Ours	LaDiCoW (CSFL)	S	5.01	38.91	30.86	25.54	37.44	51.02
Ours	LaDiCoW (All)	S	3.86	10.90	8.47	12.61	21.08	13.23
Ours	LaDiCoW (All)	P	7.99	13.57	11.82	16.25	33.59	30.55

In Table 7.2, we compare LaDiCoW to several Whisper-based baseline systems (described in §7.4). Surprisingly, the Whisper (Seg) baseline performed worse than Whisper (Utt) despite the former utilizing silver diarizations to segment each utterance into monolingual segments. Error analysis revealed that Whisper (Seg) was negatively impacted by high insertion error rates which occurred for many short (< 0.5s) segments of embedded language; we attribute the root cause of

this failure mode to noise in the silver labels resulting in slightly truncated words and Whisper's tendency to hallucinate given a noisy input. The Whisper-FT baselines performed much better in comparison. The improvement of Whisper-FT (CSFL) versus Whisper (Utt) demonstrates the general efficacy of the synthetic CSFL training data: all sets showed substantial improvements after fine-tuning except MUCS (a more technical domain and Bengali is unseen).

LaDiCoW systems performed the best overall. Comparing LaDiCoW (CSFL) with the Whisper-FT (CSFL) baseline, the improvements on the MUCS sets were the most significant: the improvement on the unseen Bengali-English language pair highlights that our language-factorized approach generalizes better across languages than the baseline approach. Comparing LaDiCoW (All) with the Whisper-FT (All) baseline, there are consistent improvements across all sets: this demonstrates that even in the setting where in-domain code-switched training data is available, **conditioning ASR on silver language diarization is more powerful than statically setting language ID at an utterance-level.**

Finally, the comparison of LaDiCoW with predicted versus silver language diarization exhibits a gap across the board; as expected the performance of LaDi limits the overall performance of LaDiCoW. If language diarization is perfect, then LaDiCoW significantly outperforms Whisper-FT baselines. However, in practice the errors in predicted language diarization are propagating errors in transcription – **currently conditioning ASR on predicted language diarization significantly degrades performance.** Next, we'll address the key follow-up question: how much better would ASR get if language diarization improved by $X\%$.

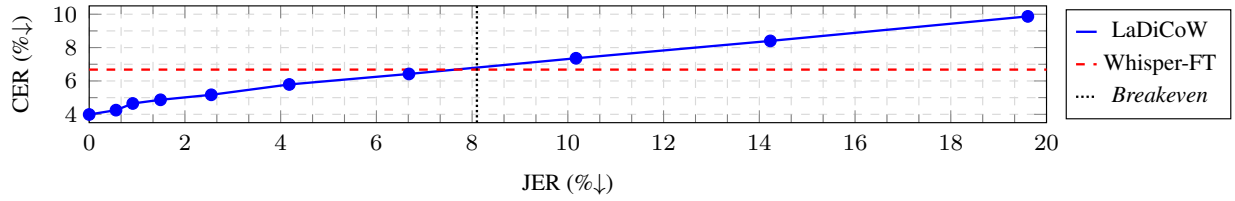

Figure 7.1: Cascaded Performance Trendline: LaDiCoW performance (CER) as a function of LaDi performance (JER).

In Figure 7.1, the left-most blue dot represents LaDiCoW given silver language diarization (JER=0) and the right-most dot represents LaDiCoW given predicted language diarization (JER=19.61). The remaining blue dots represent LaDiCoW given simulated language diarizations with linearly decreasing JER: this trend-line estimates that LaDiCoW surpasses the Whisper-FT baseline on CSFL-READ if JER is less than 8.1. This analysis suggests that **any future improvements to language diarization are expected to yield linear improvements to ASR performance.**

Breakdowns by Language

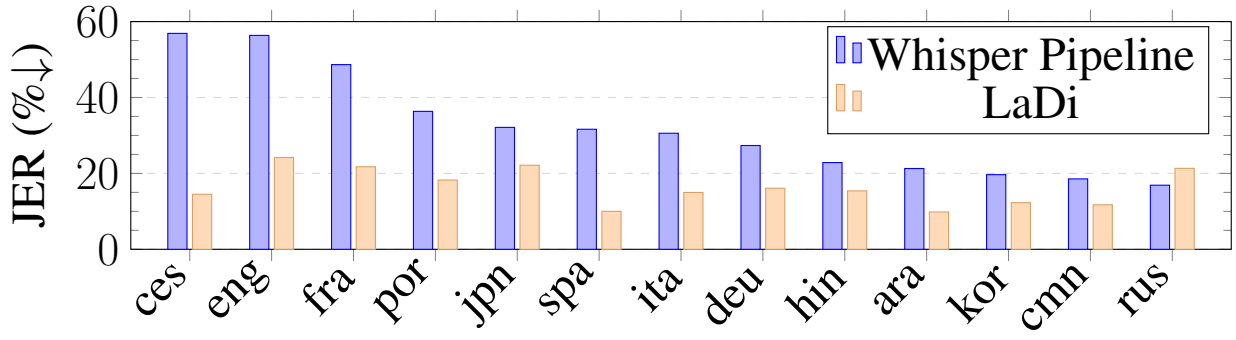

Figure 7.2: Language Diarization by Language (CSFL-READ).

In Figure 7.2, we breakdown the language diarization performance of LaDi (All) across the 13 languages (12 matrix, 1 embedded) of the CSFL-READ set. Compared to the Whisper Pipeline baseline, LaDi's performance is far more consistent across languages. Unsurprisingly English, the only embedded language, had highest JER for LaDi in this breakdown.

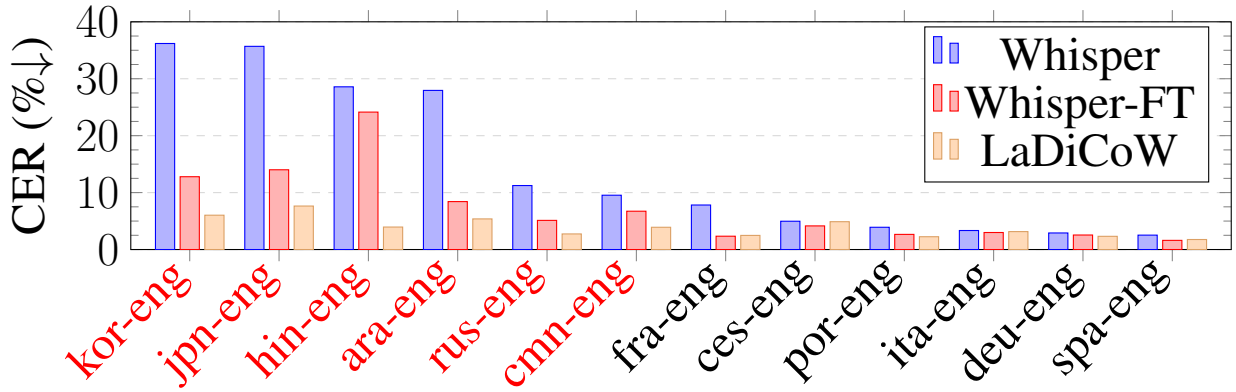

Figure 7.3: ASR by Language (CSFL-READ). Code-switched language pairs with distinct writing scripts are colored red.

In Figure 7.3, we breakdown the ASR performance of LaDiCoW (All) across the 12 code-switched language-pairs of the CSFL-READ set. Compared to the two Whisper baselines, **LaDiCoW performed significantly better on language pairs with distinct writing scripts**, highlighting the benefit of LaDiCoW's ability to switch between language-specific transcription modes within an utterance.

7.6 Conclusion

We demonstrate that by conditioning Whisper on language diarization, rather than statically setting an utterance-level language ID, code-switched transcription performance improves significantly. Our results also indicate that synthetic code-switched data can enable the construction of reasonable language diarization and language-diarization-conditioned ASR systems, but does not obviate the need for the collection of naturally occurring code-switched speech. We hope that this proof of concept affects future development towards new speech systems that can "code-switch anything".

Part III

Sequential, Non-Temporal Multi-Sequence Modeling

Chapter 8

Multi-Decoder: End-to-End Differentiable Cascaded Speech Translation

Summary

This chapter delves into **sequential, non-temporal** multi-sequence modeling which is required for speech translation with an end-to-end model.

End-to-end approaches for sequence tasks are becoming increasingly popular. Yet for complex sequence tasks, like speech translation, systems that cascade several models trained on sub-tasks have shown to be superior, suggesting that the compositionality of cascaded systems simplifies learning and enables sophisticated search capabilities. In this chapter, we present an end-to-end framework that exploits compositionality to learn *searchable* hidden representations at intermediate stages of a sequence model using decomposed sub-tasks. These hidden intermediates can be improved using beam search to enhance the overall performance and can also incorporate external models at intermediate stages of the network to re-score or adapt towards out-of-domain data. One instance of the proposed framework is a Multi-Decoder model for speech translation that extracts the *searchable hidden intermediates* from a speech recognition sub-task. The model demonstrates the aforementioned benefits and outperforms the previous state-of-the-art by around +6 and +3 BLEU on the two test sets of Fisher-CallHome and by around +3 and +4 BLEU on the English-German and English-French test sets of MuST-C.

8.1 Introduction

The principle of compositionality loosely states that a complex whole is composed of its parts and the rules by which those parts are combined (Lake and Baroni, 2018). This principle is present

in engineering, where task decomposition of a complex system is required to assess and optimize task allocations (Levis et al., 1994), and in natural language, where paragraph coherence and discourse analysis rely on decomposition into sentences (Johnson, 1992; Kuo, 1995) and sentence level semantics relies on decomposition into lexical units (Liu et al., 2020c).

Similarly, many sequence-to-sequence tasks that convert one sequence into another (Sutskever et al., 2014) can be decomposed to simpler sequence sub-tasks in order to reduce the overall complexity. For example, speech translation systems, which seek to process speech in one language and output text in another language, can be naturally decomposed into the transcription of source language audio through automatic speech recognition (ASR) and translation into the target language through machine translation (MT). Such cascaded approaches have been widely used to build practical systems for a variety of sequence tasks like hybrid ASR (Hinton et al., 2012), phrase-based MT (Koehn et al., 2007b), and cascaded ASR-MT systems for speech translation (ST) (Pham et al., 2019a).

End-to-end sequence models like encoder-decoder models (Bahdanau et al., 2015; Vaswani et al., 2017), are attractive in part due to their simplistic design and the reduced need for handcrafted features. However, studies have shown mixed results compared to cascaded models particularly for complex sequence tasks like speech translation (Inaguma et al., 2020) and spoken language understanding (Coucke et al., 2018). Although direct target sequence prediction avoids the issue of error propagation from one system to another in cascaded approaches (Tzoukermann and Miller, 2018), there are many attractive properties of cascaded systems, missing in end-to-end approaches, that are useful in complex sequence tasks.

In particular, we are interested in (1) the strong search capabilities of the cascaded systems that compose the final task output from individual system predictions (Mohri et al., 2002; Kumar et al., 2006; Beck et al., 2019), (2) the ability to incorporate external models to re-score each individual system (Och and Ney, 2002; Huang and Chiang, 2007), (3) the ability to easily adapt individual components towards out-of-domain data (Koehn and Schroeder, 2007; Peddinti et al., 2015), and finally (4) the ability to monitor performance of the individual systems towards the decomposed sub-task (Tillmann and Ney, 2003; Meyer et al., 2016).

In this paper, we seek to incorporate these properties of cascaded systems into end-to-end sequence models. We first propose a generic framework to learn *searchable hidden intermediates* using an auto-regressive encoder-decoder model for any decomposable sequence task (§8.3). We then apply this approach to speech translation, where the intermediate stage is the output of ASR, by passing continuous hidden representations of discrete transcript sequences from the ASR sub-net decoder to the MT sub-net encoder. By doing so, we gain the ability to use beam search with optional external model re-scoring on the hidden intermediates, while maintaining end-to-end differentiability. Next, we suggest mitigation strategies for the error propagation issues inherited from decomposition.

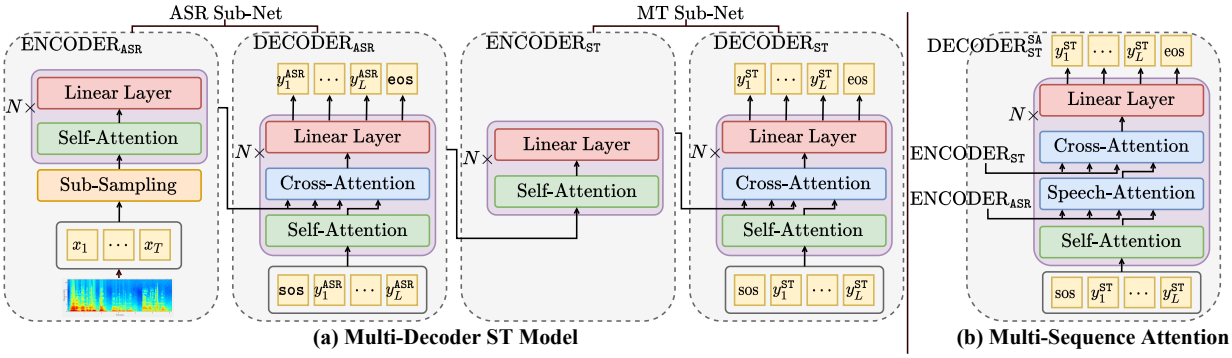

Figure 8.1: The left side present the schematics and the information flow of our proposed framework applied to ST, in a model we call the Multi-Decoder. Our model decomposes ST into ASR and MT sub-nets, each of which consist of an encoder and decoder. The right side displays a Multi-Sequence Attention variant of the $\text{DECODER}_{\text{ST}}$ that is conditioned on both speech information via the $\text{ENCODER}_{\text{ASR}}$ and transcription information via the $\text{ENCODER}_{\text{ST}}$.

We show the efficacy of *searchable intermediate representations* in our proposed model, called the Multi-Decoder, on speech translation with a 5.4 and 2.8 BLEU score improvement over the previous state-of-the-arts for Fisher and CallHome test sets respectively (§8.6). We extend these improvements by an average of 0.5 BLEU score through the aforementioned benefit of re-scoring the intermediate search with external models trained on the same dataset. We also show a method for monitoring sub-net performance using oracle intermediates that are void of search errors (§8.6). Finally, we show how these models can adapt to out-of-domain speech translation datasets, how our approach can be generalized to other sequence tasks like speech recognition, and how the benefits of decomposition persist even for larger corpora like MuST-C (§8.6).

8.2 Background and Motivation

Compositionality in Sequences Models

The probabilistic space of a sequence is combinatorial in nature, such that a sentence of L words from a fixed vocabulary $\mathcal{V}$ would have an output space $\mathcal{S}$ of size $|\mathcal{V}|^L$. In order to deal with this combinatorial output space, an output sentence is decomposed into labeled target tokens, $\mathbf{y} = (y_1, y_2, \ldots, y_L)$, where $y_l \in \mathcal{V}$.

$$P(\mathbf{y} \mid \mathbf{x}) = \prod_{i=1}^{L} P(y_i \mid \mathbf{x}, y_{1:i-1})$$

An auto-regressive encoder-decoder model uses the above probabilistic decomposition in sequence-to-sequence tasks to learn next word prediction, which outputs a distribution over the next target token y_l given the previous tokens $y_{1:l-1}$ and the input sequence $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_t, \ldots, \mathbf{x}_T)$, where T is the input sequence length. In the next sub-section we detail the training and inference of these models.

Auto-regressive Encoder-Decoder Models

Training: In an auto-regressive encoder-decoder model, the ENCODER maps the input sequence $\mathbf{x}$ to a sequence of continuous hidden representations $\mathbf{h}^E = (\mathbf{h}_1^E, \mathbf{h}_t^E, \ldots, \mathbf{h}_T^E)$, where $\mathbf{h}_t^E \in \mathbb{R}^d$. The DECODER then auto-regressively maps $\mathbf{h}^E$ and the preceding ground-truth output tokens, $\hat{y}_{1:l-1}$, to $\mathbf{h}_l^D$, where $\mathbf{h}_l^D \in \mathbb{R}^d$. The sequence of decoder hidden representations form $\mathbf{h}^D = (\mathbf{h}_1^D, \mathbf{h}_l^D, \ldots, \mathbf{h}_L^D)$ and the likelihood of each output token y_l is given by SOFTMAXOUT, which denotes an affine projection of $\mathbf{h}_l^D$ to $\mathcal{V}$ followed by a softmax function.

$$\mathbf{h}^E = \textsc{Encoder}(\mathbf{x})$$

$$\hat{\mathbf{h}}_l^D = \textsc{Decoder}(\mathbf{h}^E, \hat{y}_{1:l-1}) \tag{8.1}$$

$$P(y_l \mid \hat{y}_{1:l-1}, \mathbf{h}^E) = \textsc{SoftmaxOut}(\hat{\mathbf{h}}_l^D) \tag{8.2}$$

During training, the DECODER performs token classification for next word prediction by considering only the ground truth sequences for previous tokens $\hat{\mathbf{y}}$. We refer to this $\hat{\mathbf{h}}^D$ as *oracle* decoder representations, which will be discussed later.

Inference: During inference, we can maximize the likelihood of the entire sequence from the output space $\mathcal{S}$ by composing the conditional probabilities of each step for the L tokens in the sequence.

$$\mathbf{h}_l^D = \textsc{Decoder}(\mathbf{h}^E, y_{1:l-1}) \tag{8.3}$$

$$P(y_l \mid \mathbf{x}, y_{1:l-1}) = \textsc{SoftmaxOut}(\mathbf{h}_l^D)$$

$$\tilde{\mathbf{y}} = \underset{\mathbf{y} \in \mathcal{S}}{\operatorname{argmax}} \prod_{i=1}^{L} P(y_i \mid \mathbf{x}, y_{1:i-1}) \tag{8.4}$$

This is an intractable search problem and it can be approximated by either greedily choosing argmax at each step or using a search algorithm like beam search to approximate $\tilde{\mathbf{y}}$. Beam search (Reddy, 1988) generates candidates at each step and prunes the search space to a tractable beam size of B most likely sequences. As $B \to \infty$, the beam search result would be equivalent to

equation 8.4.

$$\textsc{GreedySearch} \coloneqq \underset{y_l}{\operatorname{argmax}}\, P(y_l \mid \mathbf{x}, y_{1:l-1})$$
$$\textsc{BeamSearch} \coloneqq \textsc{Beam}(P(y_l \mid \mathbf{x}, y_{1:l-1}))$$

In approximate search for auto-regressive models, like beam search, the DECODER receives alternate candidates of previous tokens to find candidates with a higher likelihood as an overall sequence. This also allows for the use of external models like Language Models (LM) or Connectionist Temporal Classification Models (CTC) for re-scoring candidates (Hori et al., 2017b).

8.3 Proposed Framework

In this section, we present a general framework to exploit natural decompositions in sequence tasks which seek to predict some output $\mathcal{C}$ from an input sequence $\mathcal{A}$. If there is an intermediate sequence $\mathcal{B}$ for which $\mathcal{A} \rightarrow \mathcal{B}$ sequence transduction followed by $\mathcal{B} \rightarrow \mathcal{C}$ prediction achieves the original task, then the original $\mathcal{A} \rightarrow \mathcal{C}$ task is decomposable.

In other words, if we can learn $P(\mathcal{B} \mid \mathcal{A})$ then we can learn the overall task of $P(\mathcal{C} \mid \mathcal{A})$ through $\max_{\mathcal{B}}(P(\mathcal{C} \mid \mathcal{A},\ \mathcal{B})P(\mathcal{B} \mid \mathcal{A}))$, approximated using Viterbi search. We define a first encoder-decoder $\textsc{Sub}_{\mathcal{A}\rightarrow\mathcal{B}}\textsc{Net}$ to map an input sequence $\mathcal{A}$ to a sequence of decoder hidden states, $\mathbf{h}^{D_{\mathcal{B}}}$. Then we define a subsequent $\textsc{Sub}_{\mathcal{B}\rightarrow\mathcal{C}}\textsc{Net}$ to map $\mathbf{h}^{D_{\mathcal{B}}}$ to the final probabilistic output space of $\mathcal{C}$. Therefore, we call $\mathbf{h}^{D_{\mathcal{B}}}$ *hidden intermediates*. The following equations shows the two sub-networks of our framework, $\textsc{Sub}_{\mathcal{A}\rightarrow\mathcal{B}}\textsc{Net}$ and $\textsc{Sub}_{\mathcal{B}\rightarrow\mathcal{C}}\textsc{Net}$, which can be trained end-to-end while also exploiting compositionality in sequence tasks. [1]

$\textsc{Sub}_{\mathcal{A}\rightarrow\mathcal{B}}\textsc{Net}$:

$$\begin{aligned}
\mathbf{h}^{E} &= \textsc{Encoder}_{\mathcal{A}}(\mathcal{A}) \\
\hat{\mathbf{h}}_l^{D_{\mathcal{B}}} &= \textsc{Decoder}_{\mathcal{B}}(\mathbf{h}^{E}, \hat{\mathbf{y}}_{1:l-1}^{\mathcal{B}}) \\
P(y_l^{\mathcal{B}} \mid \hat{\mathbf{y}}_{1:l-1}^{\mathcal{B}}, \mathbf{h}^{E}) &= \textsc{SoftmaxOut}(\hat{\mathbf{h}}_l^{D_{\mathcal{B}}})
\end{aligned} \tag{8.5}$$

$\textsc{Sub}_{\mathcal{B}\rightarrow\mathcal{C}}\textsc{Net}$:

$$P(\mathcal{C} \mid \hat{\mathbf{h}}_l^{D_{\mathcal{B}}}) = \textsc{Sub}_{\mathcal{B}\rightarrow\mathcal{C}}\textsc{Net}(\hat{\mathbf{h}}_l^{D_{\mathcal{B}}}) \tag{8.6}$$

[1] Note that this framework does not use locally-normalized softmax distributions but rather the hidden representations, thereby avoiding label bias issues when combining multiple sub-systems (Bottou et al., 1997; Wiseman and Rush, 2016).

Note that the final prediction, given by equation 8.6, does not need to be a sequence and can be a categorical class like in spoken language understanding tasks. Next we will show how the *hidden intermediates* become *searchable* during inference.

Searchable Hidden Intermediates

As stated in section §8.2, approximate search algorithms maximize the likelihood, $P(\mathbf{y} \mid \mathbf{x})$, of the entire sequence by considering different candidates y_l at each step. Candidate-based search, particularly in auto-regressive encoder-decoder models, also affects the decoder hidden representation, $\mathbf{h}^D$, as these are directly dependent on the previous candidate (refer to equations 8.1 and 8.3). This implies that by searching for better approximations of the previous predicted tokens, $\mathbf{y}_{l-1} = (\mathbf{y}_{\text{BEAM}})_{l-1}$, we also improve the decoder hidden representations for the next token, $\mathbf{h}^D_l = (\mathbf{h}^D_{\text{BEAM}})_l$. As $\mathbf{y}_{\text{BEAM}} \rightarrow \hat{\mathbf{y}}$, the decoder hidden representations tend to the *oracle* decoder representations that have only errors from next word prediction, $\mathbf{h}^D_{\text{BEAM}} \rightarrow \hat{\mathbf{h}}^D$. A perfect search is analogous to choosing the ground truth $\hat{y}$ at each step, which would yield $\hat{\mathbf{h}}^D$.

We apply this beam search of hidden intermediates, thereby approximating $\hat{\mathbf{h}}^{D_\mathcal{B}}$ with $\mathbf{h}^{D_\mathcal{B}}_{\text{BEAM}}$. This process is illustrated in algorithm 1, which shows beam search for $\mathbf{h}^{D_\mathcal{B}}_{\text{BEAM}}$ that are subsequently passed to the $\text{SUB}_{\mathcal{B}\rightarrow\mathcal{C}}\text{NET}$.[2] An external model like an LM or a CTC model can be used to generate an alternate sequence likelihood, $P_{\text{EXT}}(\mathbf{y}^\mathcal{B}_l)$, which can be combined with the $\text{SUB}_{\mathcal{A}\rightarrow\mathcal{B}}\text{NET}$ likelihood, $P_\mathcal{B}(\mathbf{y}^\mathcal{B}_l \mid \mathbf{x})$, with a tunable parameter λ.

We can monitor the performance of the $\text{SUB}_{\mathcal{A}\rightarrow\mathcal{B}}\text{NET}$ by comparing the decoded intermediate sequence $\mathbf{y}^\mathcal{B}_{\text{BEAM}}$ to the ground truth $\hat{\mathbf{y}}^\mathcal{B}$. We can also monitor the $\text{SUB}_{\mathcal{B}\rightarrow\mathcal{C}}\text{NET}$ performance by using the aforementioned *oracle* representations of the intermediates, $\hat{\mathbf{h}}^{D_\mathcal{B}}$, which can be obtained by feeding the ground truth $\hat{\mathbf{y}}^\mathcal{B}$ to $\text{DECODER}_\mathcal{B}$. By passing $\hat{\mathbf{h}}^{D_\mathcal{B}}$ to $\text{SUB}_{\mathcal{B}\rightarrow\mathcal{C}}\text{NET}$, we can observe its performance in a vacuum, i.e. void of search errors in the hidden intermediates.

Multi-Decoder Model

In order to show the applicability of our end-to-end framework we propose our Multi-Decoder model for speech translation. This model predicts a sequence of text translations $\mathbf{y}^{\text{ST}}$ from an input sequence of speech $\mathbf{x}$ and uses a sequence of text transcriptions $\mathbf{y}^{\text{ASR}}$ as an intermediate. In this case, the $\text{SUB}_{\mathcal{A}\rightarrow\mathcal{B}}\text{NET}$ in equation 8.5 is specified as the ASR sub-net and the $\text{SUB}_{\mathcal{B}\rightarrow\mathcal{C}}\text{NET}$ in equation 8.6 is specified as the MT sub-net. Since the MT sub-net is also a sequence prediction task, both sub-nets are encoder-decoder models in our architecture (Bahdanau et al., 2015; Vaswani et al., 2017). In Figure 8.1 we illustrate the schematics of our transformer based Multi-Decoder ST

[2]The algorithm shown only considers a single top approximation of the search; however, with added time-complexity, the final task prediction improves with the n-best $\mathbf{h}^{D_\mathcal{B}}_{\text{BEAM}}$ for selecting the best resultant $\mathcal{C}$.

Algorithm 1 Beam Search for Hidden Intermediates: We perform beam search to approximate the most likely sequence for the sub-task $\mathcal{A} \rightarrow \mathcal{B}$, $\mathbf{y}^{\mathcal{B}}_{\text{BEAM}}$, while collecting the corresponding $\text{DECODER}_{\mathcal{B}}$ hidden representations, $\mathbf{h}^{D_{\mathcal{B}}}_{\text{BEAM}}$. The output $\mathbf{h}^{D_{\mathcal{B}}}_{\text{BEAM}}$, is passed to the final sub-network to predict final output $\mathcal{C}$ and $\mathbf{y}^{\mathcal{B}}_{\text{BEAM}}$ is used for monitoring performance on predicting $\mathcal{B}$.

Initialize: BEAM $\leftarrow \{\text{sos}\}$; k $\leftarrow$ beam size
$\mathbf{h}^{E_{\mathcal{A}}} \leftarrow \text{ENCODER}_{\mathcal{A}}(\mathbf{x})$

1. **For** $l = 1$ to $\max_{\text{STEPS}}$:
   (a) **For each** $\mathbf{y}^{\mathcal{B}}_{l-1} \in$ BEAM:
      i. Compute decoder hidden state: $\mathbf{h}^{D_{\mathcal{B}}}_{l} \leftarrow \text{DECODER}_{\mathcal{B}}(\mathbf{h}^{E_{\mathcal{A}}}, \mathbf{y}^{\mathcal{B}}_{l-1})$
      ii. **For each** candidate token $\mathbf{y}^{\mathcal{B}}_{l} \in \mathbf{y}^{\mathcal{B}}_{l-1} + \{\mathcal{V}\}$:
         A. Compute score: $s_l \leftarrow P_{\mathcal{A}\rightarrow\mathcal{B}}(\mathbf{y}^{\mathcal{B}}_{l} \mid \mathbf{x})^{1-\lambda} P_{\text{EXT}}(\mathbf{y}^{\mathcal{B}}_{l})^{\lambda}$
         B. Store hypothesis: $\mathcal{H} \leftarrow (s_l, \mathbf{y}^{\mathcal{B}}_{l}, \mathbf{h}^{D_{\mathcal{B}}}_{l})$
   (b) Update beam: BEAM $\leftarrow \arg^{\text{k}}\max(\mathcal{H})$

$(s^{\mathcal{B}}, \mathbf{y}^{\mathcal{B}}_{\text{BEAM}}, \mathbf{h}^{D_{\mathcal{B}}}_{\text{BEAM}}) \leftarrow \text{argmax}(\text{BEAM})$
Return $\mathbf{y}^{\mathcal{B}}_{\text{BEAM}} \rightarrow \text{SUB}_{\mathcal{A}\rightarrow\mathcal{B}}\text{NET}$ **Return** $\mathbf{h}^{D_{\mathcal{B}}}_{\text{BEAM}} \rightarrow$ Final $\text{SUB}_{\mathcal{B}\rightarrow\mathcal{C}}\text{NET}$

model which can also be summarized as follows:

$$\mathbf{h}^{E_{\text{ASR}}} = \text{ENCODER}_{\text{ASR}}(\mathbf{x}) \tag{8.7}$$

$$\hat{\mathbf{h}}^{D_{\text{ASR}}}_{l} = \text{DECODER}_{\text{ASR}}(\mathbf{h}^{E_{\text{ASR}}}, \hat{y}^{\text{ASR}}_{1:l-1}) \tag{8.8}$$

$$\mathbf{h}^{E_{\text{ST}}} = \text{ENCODER}_{\text{ST}}(\hat{\mathbf{h}}^{D_{\text{ASR}}}) \tag{8.9}$$

$$\hat{\mathbf{h}}^{D_{\text{ST}}}_{l} = \text{DECODER}_{\text{ST}}(\mathbf{h}^{E_{\text{ST}}}, \hat{y}^{\text{ST}}_{1:l-1}) \tag{8.10}$$

As we can see from Equations 8.9 and 8.10, the MT sub-network attends only to the decoder representations, $\hat{\mathbf{h}}^{D_{\text{ASR}}}$, of the ASR sub-network, which could lead to the error propagation issues from the ASR sub-network to the MT sub-network similar to the cascade systems, as mentioned in §8.1. To alleviate this problem, we modify equation 8.10 such that $\text{DECODER}_{\text{ST}}$ attends to both $\mathbf{h}^{E_{\text{ST}}}$ and $\mathbf{h}^{E_{\text{ASR}}}$:

$$\hat{\mathbf{h}}^{D^{\text{SA}}_{\text{ST}}}_{l} = \text{DECODER}^{\text{SA}}_{\text{ST}}(\mathbf{h}^{E_{\text{ST}}}, \mathbf{h}^{E_{\text{ASR}}}, \hat{y}^{\text{ST}}_{1:l-1}) \tag{8.11}$$

We use the multi-sequence cross-attention discussed by Helcl et al. (2018), shown on the right side of Figure 8.1, to condition the final outputs generated by $\hat{\mathbf{h}}^{D_{\text{ST}}}_{l}$ on both speech and transcript information in an attempt to allow our network to recover from intermediate mistakes during inference. We call this model the Multi-Decoder w/ Speech-Attention.

8.4 Baseline Encoder-Decoder Model

For our baseline model, we use an end-to-end encoder-decoder (Enc-Dec) ST model with ASR joint training (Inaguma et al., 2020) as an auxiliarly loss to the speech encoder. In other words, the model consumes speech input using the $\text{ENCODER}_{\text{ASR}}$, to produce $\mathbf{h}^{E_{\text{ASR}}}$, which is used for cross-attention by $\text{DECODER}_{\text{ASR}}$ and the $\text{DECODER}_{\text{ST}}$. Using the decomposed ASR task as an auxiliary loss also helps the baseline Enc-Dec model and provide strong baseline performance, as we will see in Section 8.6.

8.5 Data and Experimental Setup

Data: We demonstrate the efficacy of our proposed approach on ST in the Fisher-CallHome corpus (Post et al., 2013) which contains 170 hours of Spanish conversational telephone speech, transcriptions, and English translations. All punctuations except apostrophes were removed and results are reported in terms of detokenized case-insensitive BLEU (Papineni et al., 2002; Post, 2018). We compute BLEU using the 4 references in Fisher (dev, dev2, and test) and the single reference in CallHome (dev and test) (Post et al., 2013; Kumar et al., 2014; Weiss et al., 2017). We use a joint source and target vocabulary of 1K byte pair encoding (BPE) units (Kudo and Richardson, 2018a).

We prepare the corpus using the ESPnet library and we follow the standard data preparation, where inputs are globally mean-variance normalized log-mel filterbank and pitch features from up-sampled 16kHz audio (Watanabe et al., 2018). We also apply speed perturbations of 0.9 and 1.1 and the SS SpecAugment policy (Park et al., 2019).

Baseline Configuration: All of our models are implemented using the ESPnet library and trained on 3 NVIDIA Titan 2080Ti GPUs for $\approx$12 hours. For the Baseline Enc-Dec baseline, discussed in §8.4, we use an $\text{ENCODER}_{\text{ASR}}$ consisting of a convolutional sub-sampling by a factor of 4 (Watanabe et al., 2018) and 12 transformer encoder blocks with 2048 feed-forward dimension, 256 attention dimension, and 4 attention heads. The $\text{DECODER}_{\text{ASR}}$ and $\text{DECODER}_{\text{ST}}$ both consist of 6 transformer decoder blocks with the same configuration as $\text{ENCODER}_{\text{ASR}}$. There are 37.9M trainable parameters. We apply dropout of 0.1 for all components, detailed in the Appendix (8.9).

We train our models using an effective batch-size of 384 utterances and use the Adam optimizer (Kingma and Ba, 2015) with inverse square root decay learning rate schedule. We set learning rate to 12.5, warmup steps to 25K, and epochs to 50. We use joint training with hybrid CTC/attention ASR (Watanabe et al., 2017b) by setting mtl-alpha to 0.3 and asr-weight to 0.5 as defined by Watanabe et al. (2018). During inference, we perform beam search (Seki et al., 2019b) on the ST

sequences, using a beam size of 10, length penalty of 0.2, max length ratio of 0.3 (Watanabe et al., 2018).

Multi-Decoder Configuration: For the Multi-Decoder ST model, discussed in §8.3, we use the same transformer configuration as the baseline for the $\text{ENCODER}_{\text{ASR}}$, $\text{DECODER}_{\text{ASR}}$, and $\text{DECODER}_{\text{ST}}$. Additionally, the Multi-Decoder has an $\text{ENCODER}_{\text{ST}}$ consisting of 2 transformer encoder blocks with the same configuration as $\text{ENCODER}_{\text{ASR}}$, giving a total of 40.5M trainable parameters. The training configuration is also the same as for the baseline. For the Multi-Decoder w/ Speech-Attention model (42.1M trainable parameters), we increase the attention dropout of the ST decoder to 0.4 and dropout on all other components of the ST decoder to 0.2 while keeping dropout on the remaining components at 0.1. We verified that increasing the dropout does not help the vanilla multi-decoder ST model.

During inference, we perform beam search on both the ASR and ST output sequences, as discussed in §8.3. The ST beam search is identical to that of the baseline. For the intermediate ASR beam search, we use a beam size of 16, length penalty of 0.2, max length ratio of 0.3. In some of our experiments, we also include fusion of a source language LM with a 0.2 weight and CTC with a 0.3 weight to re-score the intermediate ASR beam search (Watanabe et al., 2017b). For the Speech-Attention variant, we increase LM weight to 0.4.

Note that the ST beam search configuration remains constant across our baseline and Multi-Decoder experiments as our focus is on improving overall performance through searchable intermediate representations. Thus, the various re-scoring techniques applied to the ASR beam search are options newly enabled by our proposed architecture and are not used in the ST beam search.

8.6 Results

Table 8.1 presents the overall ST performance (BLEU) of our proposed Multi-Decoder model. Our model improves by +2.9/+0.3 (Fisher/CallHome) over the best cascaded baseline and by +5.6/+1.5 BLEU over the best published end-to-end baselines. With Speech-Attention, our model improves by +3.4/+1.6 BLEU over the cascaded baselines and +7.1/+2.8 BLEU over encoder-decoder baselines. Both the Multi-Decoder and Multi-Decoder w/ Speech-Attention on average are further improved by +0.9/+0.4 BLEU through ASR re-scoring.[3]

Table 8.1 also includes our implementation of the Baseline Enc-Dec model discussed in §8.4. In this way, we are able to make a fair comparison with our framework as we control the model and inference configurations to be analagous. For instance, we keep the same search parameters for the

[3]We also evaluate our models using other MT metrics to supplement these results, as shown in the Appendix (8.9).

Model Type	Model Name	Uses Speech Transcripts	Fisher			CallHome	
			dev(↑)	dev2(↑)	test(↑)	dev(↑)	test(↑)
Cascade	Inaguma et al. (2020)	✓	41.5	43.5	42.2	**19.6**	**19.8**
Cascade	ESPnet ASR+MT (2018)	✓	**50.4**	**51.2**	**50.7**	**19.6**	19.2
Enc-Dec	Weiss et al. (2017) ♢	✗	46.5	47.3	47.3	16.4	16.6
Enc-Dec	Weiss et al. (2017) ♢	✓	48.3	49.1	48.7	16.8	17.4
Enc-Dec	Inaguma et al. (2020)	✓	46.6	47.6	46.5	16.8	16.8
Enc-Dec	Guo et al. (2021)	✓	48.7	49.6	47.0	18.5	**18.6**
Enc-Dec	Our Implementation	✓	**49.6**	**50.9**	**49.5**	**19.1**	18.2
Multi-Decoder	Our Proposed Model	✓	52.7	53.3	52.6	20.5	20.1
Multi-Decoder	+ASR Re-scoring	✓	53.3	54.2	53.7	21.1	20.8
Multi-Decoder	+Speech-Attention	✓	**54.6**	**54.6**	**54.1**	**21.7**	**21.4**
Multi-Decoder	+ASR Re-scoring	✓	**55.2**	**55.2**	**55.0**	**21.7**	**21.5**

Table 8.1: Results presenting the overall performance (BLEU) of our proposed multi-decoder model. Cascade and Enc-Dec results from previous papers and our own implementation of the Enc-Dec are shown for comparison. The best performing models are **highlighted**. ♢Implemented with LSTM, while all others are Transformer-based.

final output in the baseline and the Multi-Decoder to demonstrate impact of the intermediate beam search.

Benefits

Sub-network Performance Monitoring

An added benefit of our proposed approach over the Baseline Enc-Dec is the ability to monitor the individual performances of the ASR (% WER) and MT (BLEU) sub-nets as shown in Table 8.2. The Multi-Decoder w/ Speech-Attention shows a greater MT sub-net performance than the Multi-Decoder as well as a slight improvement of the ASR sub-net, suggesting that ST can potentially help ASR.

Beam Search for Better Intermediates

The overall ST performance improves when a higher beam size is used in the intermediate ASR search, and this increase can be attributed to the improved ASR sub-net performance. Figure 1 shows this trend across ASR beam sizes of 1, 4, 8, 10, 16 while fixing the ST decoding beam size to 10. A beam size of 1, which is a greedy search, results in lower ASR sub-net and overall ST

Model	Overall ST(↑)	Sub-Net ASR(↓)	Sub-Net MT(↑)
Multi-Decoder	52.7	22.6	64.9
+Speech-Attention	54.6	22.4	66.6

Table 8.2: Results presenting the overall ST performance (BLEU) of our Multi-Decoder models, along with their sub-net ASR (% WER) and MT (BLEU) performances. All results are from the Fisher dev set.

performances. As beam sizes become larger, gains taper off as can be seen between beam sizes of 10 and 16.

External Models for Better Search

External models like CTC acoustic models and language models are commonly used for re-scoring encoder-decoder models (Hori et al., 2017b), due to the difference in their modeling capabilities. CTC directly models transcripts while being conditionally independent on the other outputs given the input, and LMs predict the next token in a sequence.

Both variants of the Multi-Decoder improve due to improved ASR sub-net performance using external CTC and LM models for re-scoring, as shown in Table 8.3. We use a recurrent neural network LM trained on the Fisher-CallHome Spanish transcripts with a dev perplexity of 18.8 and the CTC model from joint loss applied during training. Neither external model incorporates additional data. Although the impact of the LM-only re-scoring is not shown in the ASR % WER, it reduces substitution and deletion rates in the ASR and this is observed to help the overall ST performance.

Error Propagation Avoidance

As discussed in §8.3, our Multi-Decoder model inherits the error propagation issue as can be seen in Figure 8.3. For the easiest bucket of utterances with $< 40\%$ WER in Multi-Decoder's ASR sub-net, our model's ST performance, as measured by the corpus BLEU of the bucket, exceeds that of the Baseline Enc-Dec. The inverse is true for the more difficult bucket of $[40, 80)\%$, showing that error propagation is limiting the performance of our model; however, we show that multi-sequence attention can alleviate this issue. For extremely difficult utterances in the $\geq 80\%$ bucket, ST performance for all three approaches is suppressed. We also provide qualitative examples of error propagation avoidance in the Appendix (8.9).

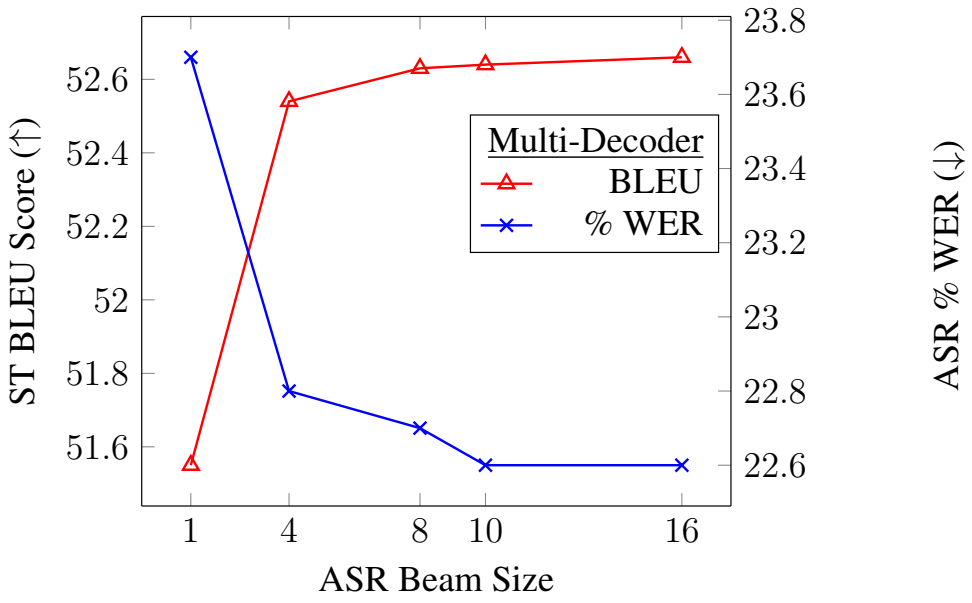

Figure 8.2: Results studying the effect of the different ASR beam sizes in the intermediate representation search on the overall ST performance (BLEU) and the ASR sub-net performance (% WER) for our multi-decoder model. Beam of 1 is same as greedy search.

Generalizability

In this section, we discuss the generalizability of our framework towards out-of-domain data. We also extend our Multi-Decoder model to other sequence tasks like speech recognition. Finally, we apply our ST models to a larger corpus with more language pairs and a different domain of speech.

Robustness through Decomposition

Like cascaded systems, searchable intermediates provide our model adaptability in individual subsystems towards out-of-domain data using external in-domain language model, thereby giving access to more in-domain data. Specifically for speech translation systems, this means we can use in-domain language models in both source and target languages. We test the robustness of our Multi-Decoder model trained on Fisher-CallHome conversational speech dataset on read speech CoVost-2 dataset (Wang et al., 2020b). In Table 8.4 we show that re-scoring the ASR sub-net with an in-domain LM improves ASR with around 10.0% lower WER, improving the overall ST performance by around +2.5 BLEU. Compared to an in-domain ST baseline (Wang et al., 2020a), our out-of-domain Multi-Decoder with in-domain ASR re-scoring demonstrates the robustness of our approach.

Model	Overall ST(↑)	Sub-Net ASR(↓)
Multi-Decoder	52.7	22.6
+ASR Re-scoring w/ LM	53.2	22.6
+ASR Re-scoring w/ CTC	52.8	22.1
+ASR Re-scoring w/ LM	**53.3**	**21.7**
Multi-Decoder w/ Speech-Attn.	54.6	22.4
+ASR Re-scoring w/ LM	55.1	22.4
+ASR Re-scoring w/ CTC	54.7	22.0
+ASR Re-scoring w/ LM	**55.2**	**21.9**

Table 8.3: Results presenting the overall ST performance (BLEU) and the sub-net ASR (% WER) of our Multi-Decoder models with external CTC and LM re-scoring in the ASR intermediate representation search. All results are from the Fisher dev set.

Decomposing Speech Transcripts

We apply our generic framework to another decomposable sequence task, speech recognition, and show the results of various levels of decomposition in Table 8.5. We show that with phoneme, character, or byte-pair encoding (BPE) sequences as intermediates, the Multi-Decoder presents strong results on both Fisher and CallHome test sets. We also observe that the BPE intermediates perform better than phoneme/character variants, which could be attributed to the reduced search capabilities of encoder-decoder models using beam search on longer sequences (Sountsov and Sarawagi, 2016) like in phoneme/character sequences.

Extending to MuST-C Language Pairs

In addition to our results using the 170 hours of the Spanish-English Fisher-CallHome corpus, in Table 8.6 we show that our decompositional framework is also effective on larger ST corpora. In particular, we use 400 hours of English-German and 500 hours of English-French ST from the MuST-C corpus (Di Gangi et al., 2019). Our Multi-Decoder model improves by +2.7 and +1.5 BLEU, in German and French respectively, over end-to-end baselines from prior works that do not use additional training data. We show that ASR re-scoring gives an additional +0.1 and +0.4 BLEU improvement. [4]

By extending our Multi-Decoder models to this MuST-C study, we show the generalizability of our approach across several dimensions of ST tasks. First, our approach consistently improves over

[4]Details of the MuST-C data preparation and model parameters are detailed in Appendix (8.9).

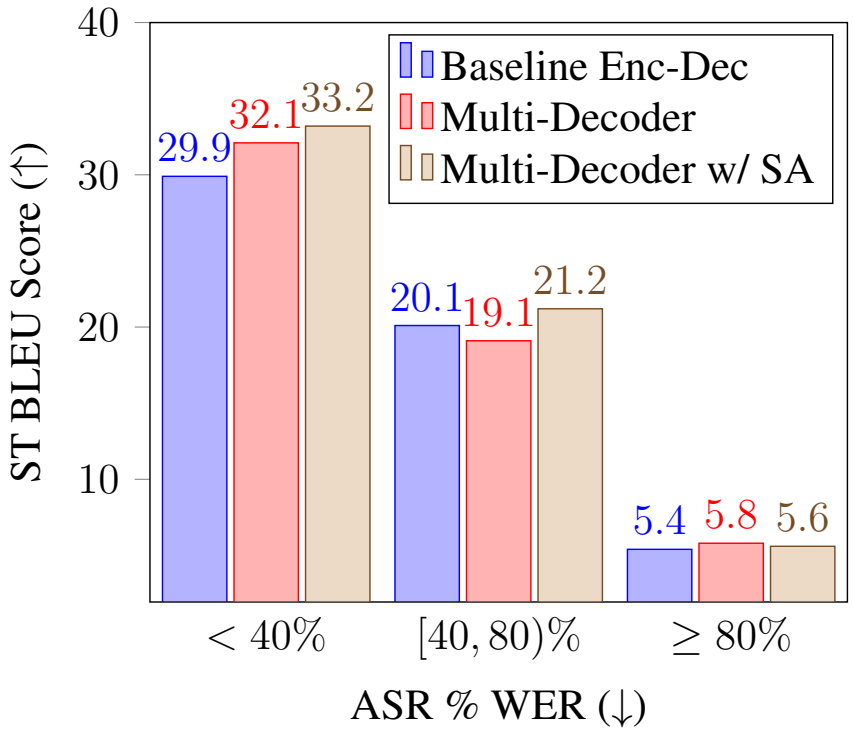

Figure 8.3: Results comparing the ST performances (BLEU) of our Baseline Enc-Dec, Multi-Decoder, and Multi-Decoder w/ Speech-Attention across different ASR difficulties measured using % WER on the Fisher dev set (1-ref). The buckets on the x-axis are determined using the utterance level % WER using the Multi-Decoder ASR sub-net performance.

baselines across multiple language-pairs. Second, our approach is robust to the distinct domains of telephone conversations from Fisher-CallHome and the TED-Talks from MuST-C. Finally, by scaling from 170 hours of Fisher-CallHome data to 500 hours of MuST-C data, we show that the benefits of decomposing sequence tasks with searchable hidden intermediates persist even with more data.

Furthermore, the performance of our Multi-Decoder models trained with only English-German or English-French ST data from MuST-C is comparable to other methods which incorporate larger external ASR and MT data in various ways. For instance, Zheng et al. (2021) use 4700 hours of ASR data and 2M sentences of MT data for pretraining and multi-task learning. Similarly, Bahar et al. (2021) use 2300 hours of ASR data and 27M sentences of MT data for pretraining. Our competitive performance without the use of any additional data highlights the data-efficient nature of our proposed end-to-end framework as opposed to the baseline encoder-decoder model, as pointed out by Sperber and Paulik (2020).

Model	Overall ST(↑)	Sub-Net ASR(↓)
IN-DOMAIN ST MODEL		
Baseline (Wang et al., 2020b)	12.0	-
+ASR Pretrain (Wang et al., 2020b) ♢	23.0	16.0
OUT-OF-DOMAIN ST MODEL		
Multi-Decoder	11.8	46.8
+ASR Re-scoring w/ in-domain LM	14.4	**36.7**
Multi-Decoder w/ Speech-Attention	12.6	46.5
+ASR Re-scoring w/ in-domain LM	**15.0**	**36.7**

Table 8.4: Results presenting the overall ST performance (BLEU) and the sub-net ASR (% WER) of our Multi-Decoder models when tested on out-of-domain data. All models were trained on the Fisher-CallHome Es→En corpus and tested on CoVost2 Es→En corpus. ♢Pretrained with 364 hours of in-domain ASR data.

8.7 Discussion and Relation to Prior Work

Compositionality: A number of recent works have constructed composable neural network modules for tasks such as visual question answering (Andreas et al., 2016), neural MT (Raunak et al., 2019), and synthetic sequence-to-sequence tasks (Lake, 2019). Modules that are first trained separately can subsequently be tightly integrated into a single end-to-end trainable model by passing differentiable soft decisions instead of discrete decisions in the intermediate stage (Bahar et al., 2021). Further, even a single encoder-decoder model can be decomposed into modular components where the encoder and decoder modules have explicit functions (Dalmia et al., 2019).

Joint Training with Sub-Tasks: End-to-end sequence models been shown to benefit from introducing joint training with sub-tasks as auxiliary loss functions for a variety of tasks like ASR (Kim et al., 2017), ST (Salesky et al., 2019; Liu et al., 2020b; Dong et al., 2020; Le et al., 2020a), SLU (Haghani et al., 2018). They have been shown to induce structure (Belinkov et al., 2020) and improve the model performance (Toshniwal et al., 2017), but this joint training may reduce data efficiency if some sub-nets are not included in the final end-to-end model (Sperber et al., 2019; Wang et al., 2020c). Our framework avoids this sub-net waste at the cost of computational load during inference.

Model	Intermediate	Fisher ASR(↓)	CallHome ASR(↓)
Enc-Dec ♢	-	23.2	45.3
Multi-Decoder	Phoneme	20.7	40.0
Multi-Decoder	Character	20.4	39.9
Multi-Decoder	BPE100	**19.7**	**38.9**

Table 8.5: Results presenting the % WER ASR performance when using the Multi-Decoder model on decomposed ASR task with phoneme, character, and BPE100 as intermediates. All results are from the Fisher-CallHome Spanish corpus. ♢(Weiss et al., 2017)

Speech Translation Decoders: Prior works have used ASR/MT decoding to improve the overall ST decoding through synchronous decoding (Liu et al., 2020b), dual decoding (Le et al., 2020a), and successive decoding (Dong et al., 2020). These works partially or fully decode ASR transcripts and use discrete intermediates to assist MT decoding. Tu et al. (2017) and Anastasopoulos and Chiang (2018) are closest to our multi-decoder ST model, however the benefits of our proposed framework are not entirely explored in these works.

Two-Pass Decoding: Two-pass decoding involves first predicting with one decoder and then re-evaluating with another decoder (Geng et al., 2018; Sainath et al., 2019; Hu et al., 2020; Rijhwani et al., 2020). The two decoders iterate on the same sequence, so there is no decomposition into sub-tasks in this method. On the other hand, our approach provides the subsequent decoder with a more structured representation than the input by decomposing the complexity of the overall task. Like two-pass decoding, our approach provides a sense of the future to the second decoder which allows it to correct mistakes from the previous first decoder.

Auto-Regressive Decoding: As auto-regressive decoders inherently learn a language model along with the task at hand, they tend to be domain specific (Samarakoon et al., 2018; Müller et al., 2020). This can cause generalizability issues during inference (Murray and Chiang, 2018a; Yang et al., 2018), impacting the performance of both the task at hand and any downstream tasks. Our approach alleviates these problems through intermediate search, external models for intermediate re-scoring, and multi-sequence attention.

Model	En→De ST(↑)	En→Fr ST(↑)
NeurST (Zhao et al., 2020)	22.9	33.3
Fairseq S2T (Wang et al., 2020a)	22.7	32.9
ESPnet-ST (Inaguma et al., 2020)	22.9	32.7
Dual-Decoder (Le et al., 2020a)	23.6	33.5
Multi-Decoder w/ Speech-Attn.	26.3	37.0
+ASR Re-scoring	**26.4**	**37.4**

Table 8.6: Results presenting the overall ST performance (BLEU) of our Multi-Decoder w/ Speech-Attention models with ASR re-scoring across two language-pairs, English-German (En→De) and English-French (En→Fr). All results are from the MuST-C tst-COMMON sets. All models use speech transcripts.

8.8 Conclusion and Future Work

We present searchable hidden intermediates for end-to-end models of decomposable sequence tasks. We show the efficacy of our Multi-Decoder model on the Fisher-CallHome Es→En and MuST-C En→De and En→Fr speech translation corpora, achieving state-of-the-art results. We present various benefits in our framework, including sub-net performance monitoring, beam search for better hidden intermediates, external models for better search, and error propagation avoidance. Further, we demonstrate the flexibility of our framework towards out-of-domain tasks with the ability to adapt our sequence model at intermediate stages of decomposition. Finally, we show generalizability by training Multi-Decoder models for the speech recognition task at various levels of decomposition.

We hope insights derived from our study stimulate research on tighter integrations between the benefits of cascaded and end-to-end sequence models. Exploiting searchable intermediates through beam search is just the tip of the iceberg for search algorithms, as numerous approximate search techniques like diverse beam search (Vijayakumar et al., 2018) and best-first beam search (Meister et al., 2020) have been recently proposed to improve diversity and approximation of the most-likely sequence. Incorporating differentiable lattice based search (Hannun et al., 2020) can also allow the subsequent sub-net to digest n-best representations.

We've now established the efficacy of sequential, non-temporal multi-sequence modeling for speech translation on a number of high resourced language pairs. Next, we consider a more resource constrained setting.

Model / Source	ASR Output	ST Output
Ground-Truth	... porque tengo a mis dos hijos acá	... because i have my two children here
Multi-Decoder	... porque tengo mis dos hijos acá	... because i have two kids here
+Speech-Attention	... porque tengo mis dos hijos acá	... because i have my two children here
Ground-Truth	puedes ayudar para que se haga justicia más rápido	you can help so that justice is served quickly
Multi-Decoder	puedes ayudar para que sea justicia más rápido	you can help so it's faster
+Speech-Attention	puedes ayudar para que sea justicia más rápido	you can help so that it's faster justice
Ground-Truth	pero tiene muchas cosas muy bonitas	but there are many beautiful things
Multi-Decoder	pero tienen muchas cosas muy bonitas	but they have a lot of nice things
+Speech-Attention	pero tienen muchas cosas muy bonitas	but there are many very beautiful things
Ground-Truth	acampar ir a pescar y ir a las montañas a esquiar	camping and fishing and going to the mountains to ski
Multi-Decoder	acampar y a pescar y y de las montañas esquiar	camping and fishing and and the mountains skiing
+Speech-Attention	a campar y ir a pescar y ir a las montañas a esquiar	camping and go fishing and go to the mountains to ski

Table 8.7: Examples where the Multi-Decoder and Multi-Decoder w/ Speech-Attention models make errors in the ASR portion of Spanish-English ST. In these cases the Speech-Attention component alleviates ASR error propagation, producing correct translations despite mistakes in transcription. Words that are transcribed/translated correctly are highlighted in green and those that are incorrect are in pink.

8.9 Appendix

Training and Inference hyperparameters

We tune training and inference hyperparameters using only the dev sets. We first determined the best hyperparameters for our baseline Enc-Dec implementation and fixed all settings not pertaining to the unique searchable hidden intermediates of our Multi-Decoder. Then, we find the best hyperparameters for our proposed models under these constraints to demonstrate a true comparison against the baseline. For our Speech-Attention variant, we found that increasing attention dropout in the ST sub-net decoder to 0.4 improved performance, which we verified was not true for the vanilla Multi-Decoder model. For our external model re-scoring, we found that a CTC weight of 0.3 is best for all Multi-Decoder and Multi-Decoder w/ Speech-Attention. The best LM weight for the Multi-Decoder was 0.2, while the best LM weight for the Multi-Decoder w/ Speech-Attention was 0.4. For both of these re-scoring hyperparameters, we tried $[0.2, 0.3, 0.4]$. For deciding the beam size, we use the experiment demonstrated in Figure 8.2 which uses beam sizes of $[1, 4, 8, 10, 16]$.

	Fisher test			CallHome test		
Model	BLEU (↑)	METEOR(↑)	TER(↓)	BLEU (↑)	METEOR(↑)	TER(↓)
Baseline Enc-Dec	49.5	37.9	42.7	18.2	22.9	68.7
Multi-Decoder	52.6	39.7	40.5	20.1	24.6	66.5
+ASR Re-scoring	53.7	40.0	39.6	20.8	24.9	65.3
+Speech-Attention	54.1	40.2	39.2	21.4	25.2	65.3
+ASR Re-scoring	**55.0**	**40.4**	**38.5**	**21.5**	**25.4**	**64.2**

Table 8.8: Results presenting the performance of our Baseline Enc-Dec implementation and our Multi-Decoder models as evaluated by three metrics: BLEU, METEOR, and Translation Edit Rate (TER). These are the same models as in Table 8.1, which uses BLEU. All results are from the Fisher-CallHome Spanish-English test corpus.

Multi-Decoder ST Performance Across Other Automatic MT Metrics

To supplement our overall ST results on the Fisher/CallHome corpus in Table 8.1, which shows BLEU scores, we also evaluated the same Multi-Decoder and Baseline Enc-Dec (Our Implementation) models on two additional metrics: METEOR (Banerjee and Lavie, 2005) and Translation Edit Rate (TER) (Snover et al., 2006). Performance across all three metrics show consistent trends, with the Multi-Decoder outperforming the Baseline Enc-Dec model on all metrics. We see that both the Multi-Decoder and Multi-Decoder w/ Speech-Attention models are improved through ASR Re-scoring. Further, the models with Speech-Attention perform better than those without.

Qualitative Examples of Error Propagation Avoidance

To supplement our qualitative analysis of the error propagation avoidance of the Multi-Decoder with Speech-Attention model in §8.6, we also show four qualitative examples in Table 8.7. In the first three examples, the Multi-Decoder and Multi-Decoder with Speech-Attention models both make the same mistakes in the ASR portion of Spanish-English translation, but the model with Speech-Attention recovers by producing correct English translations despite mistakes in the Spanish transcription. On the other hand, the model without Speech-Attention propagates the Spanish transcription errors into English translation errors. In the fourth example only the Multi-Decoder w/ Speech-Attention makes a mistake in Spanish transcription, but the English translation still recovers.

MuST-C Data Setup and Model Details

Data: We extend our approach to other language pairs from the MuST-C speech translation corpus (Di Gangi et al., 2019). These are recordings of TED talks in English with translations in various target languages. In our experiments we show results on two language pairs, namely, English-German and English-French. We use the provided dev set for deciding the training and inference hyperparameters, as mentioned in Appendix (8.9). We report detokenized case-sensitive BLEU (Post, 2018) on the tst-COMMON set. We apply the same text processing as done in (Inaguma et al., 2020) and use a joint source and target vocabulary of 8K byte pair encoding (BPE) units (Kudo and Richardson, 2018a). Similar to §8.5, we use the ESPnet library to prepare the corpus, and apply the same data preparation and augmentations.

Multi-Decoder Configuration: For the MuST-C experiments, we scaled our Multi-Decoder w/ Speech-Attention config from the Fisher-CallHome experiments by increasing the $\text{ENCODER}_{\text{ST}}$ to contain 4 transformer encoder blocks. We increased the attention dim and attention heads of the $\text{ENCODER}_{\text{ASR}}$ and $\text{DECODER}_{\text{ASR}}$ to 512 dimension and 8 heads respectively, while only increasing the attention dimension to 512 for $\text{ENCODER}_{\text{ST}}$ and $\text{DECODER}_{\text{ST}}$. This increased the total trainable parameters to 135M, which we trained on 4 NVIDIA V-100 GPUs for $\approx$3 days. We also found that increasing the attention dropout of ASR decoder to 0.2 helped with the increased parameters. We kept the remaining dropout parameters the same as our previous experiments. We also keep the remaining training configurations the same like the effective batch-size, learning rate and warmup steps, loss weighting and SpecAugment policy.

During inference, we use the same beam sizes from our Fisher-CallHome experiments and we perform a search across the length penalty and max length ratio settings using the MuST-C dev sets. In the intermediate ASR beam search we use a length penalty of 0.1 and 0.2 for English-German and English-French respectively. In the ST beam search we use a max length ratio of 0.3 and length penalties of 0.6 and 0.5 for English-German and English-French respectively. For our experiments with ASR re-scoring, we use a LM weight of 0.1 and a CTC weight of 0.1. In these re-scoring experiments we also set the ASR length penalty to 0.6 and the ST length penalty to 0.5, while increasing the ST max length ratio to 0.5. The LMs used were trained on the English transcripts of the MuST-C English-German and English-French corpora, with dev perplexities of 32.7 and 23.2 respectively.

Chapter 9

Application of *Multi-Decoder* to Dialectal Speech Translation

Summary

This chapter applies the Multi-Decoder approach introduced in Chapter 6, a **sequential, non-temporal** multi-sequence model, to a resource constrained setting.

This chapter describes CMU's submissions to the IWSLT 2022 dialect speech translation (ST) shared task for translating Tunisian-Arabic speech to English text. We use additional paired Modern Standard Arabic data (MSA) to directly improve the speech recognition (ASR) and machine translation (MT) components of our cascaded systems. We also augment the paired ASR data with pseudo translations via sequence-level knowledge distillation from an MT model and use these artificial triplet ST data to improve our end-to-end (E2E) systems. Our E2E models are based on the Multi-Decoder architecture with searchable hidden intermediates. We extend the Multi-Decoder by orienting the speech encoder towards the target language by applying ST supervision as hierarchical connectionist temporal classification (CTC) multi-task. During inference, we apply joint decoding of the ST CTC and ST autoregressive decoder branches of our modified Multi-Decoder. Finally, we apply ROVER voting, posterior combination, and minimum bayes-risk decoding with combined N-best lists to ensemble our various cascaded and E2E systems. Our best systems reached 20.8 and 19.5 BLEU on test2 (blind) and test1 respectively. Without any additional MSA data, we reached 20.4 and 19.2 on the same test sets.

9.1 Introduction

In this chapter, we present CMU's Tunisian-Arabic to English ST systems submitted to the IWSLT 2022 dialectal ST track (Anastasopoulos et al., 2022). One of our goals is to investigate dialectal transfer from large MSA ASR and MT corpora to improve Tunisian-Arabic ST performance. We also view this task as setting for extending the sequence-level knowledge distillation (SeqKD) (Kim and Rush, 2016a), E2E Multi-Decoder architecture (Dalmia et al., 2021b), and system combination methods in our IWSLT 2021 offline ST systems (Inaguma et al., 2020).

In particular, our contributions are the following:

1. Dialectal transfer from large paired MSA corpora to improve ASR and MT systems (§9.3)
2. MT SeqKD on MSA ASR data for artificial ST triplets to improve E2E ST systems (§9.3)
3. Multi-Decoder with hierarchical CTC training for target-oriented speech encodings (§9.3)
4. Multi-Decoder with CTC beam search hypothesis re-scoring during ST inference (§9.3)
5. Multi-Decoder with surface and posterior-level guidance from external models (§9.3)
6. Joint minimum bayes-risk decoding as an ensembling method (§9.3)

Results on the blind test set, test2, and ablations on the provided test set, test1, demonstrate the overall efficacy of our systems and the relative contributions of the aforementioned techniques (§9.5).

9.2 Task Description and Data Preparation

The Arabic language is not a monolith. Of its estimated 400 million native speakers, many speak in colloquial dialects such as, Tunisian-Arabic, that have relatively less standard orthographic rules and smaller ASR and MT corpora compared to formal MSA (Hussein et al., 2022). Both of these realities present challenges to building effective ST systems, and as such the dialectal speech translation shared task is an important venue for tackling these research problems.

Table 9.1 shows the corpora relevant to the shared task. The IWSLT22-Dialect corpus consists of ST triplets where 160 hours of 8kHz conversational Tunisian-Arabic speech are annotated with transcriptions and also translated into English. The MGB2 corpus (Ali et al., 2016) consists of 1100 hours of 16kHz broadcast MSA speech and the corresponding transcriptions. The OPUS corpus (Tiedemann et al., 2020) consists of 42M MSA-English translation pairs across several domains. Any systems that use MGB2 or OPUS data for pre-training, fine-tuning, or any other purpose are designated as *dialect transfer* systems.[1]

[1]We do not use self-supervised representations, morphological analyzers, or any other resources reliant on data other than the three aforementioned corpora.

	#Hours	#Sentence	
	of Speech	Arabic	English
IWSLT22-Dialect	160	0.2M	0.2M
MGB2	1100	1.1M	-
OPUS	-	42M	42M

Table 9.1: Statistics for the three corpora included in the IWSLT 2022 dialect ST shared task. IWSLT22-Dialect has triplets of speech, source Arabic transcription, and target English translation. MGB2 and OPUS have only pairs for ASR and MT respectively.

Following the shared task guidelines, punctuation is removed and English text is lower-cased. Buckwalter one-to-one transliteration of Arabic text (Habash et al., 2007) was applied to help non-Arabic speakers with ASR output interpretation. English sentences were tokenized with the `tokenizer.perl` script in the Moses toolkit (Koehn et al., 2007b) for training and detokenized for scoring. Language-specific sentence-piece vocabularies were created using the byte pair encoding (BPE) algorithm (Sennrich et al., 2016) with the `sentencepiece` toolkit.[2] Speech data was up-sampled by a factor of 3 using 0.9 and 1.1 speed perturbation ratios (Ko et al., 2015). The IWSLT22-Dialect data was upsampled to 16kHz for consistency using the `sox` toolkit[3].

9.3 Proposed Methods

In this section, we describe our cascaded (§9.3) and E2E systems (§9.3). Then we describe methods for integrating both approaches §9.3.

Cascaded ASR→MT Systems

ASR

To train ASR models for our cascaded system, we use the ESPnet (Watanabe et al., 2018) framework. Our ASR architecture is based on hybrid CTC/attention approach (Watanabe et al., 2017b) with a Conformer encoder (Gulati et al., 2020). The Conformer, which employs convolutions to model local patterns and self-attention to model long-range context, has shown to be effective on both ASR and E2E ST tasks (Guo et al., 2021; Inaguma et al., 2020). We also use a bidirectional LSTM (Hochreiter and Schmidhuber, 1997; Graves and Schmidhuber, 2005) language model (LM)

[2] `https://github.com/google/sentencepiece`
[3] `http://sox.sourceforge.net`

to re-score beam search hypotheses during inference. We ensemble multiple ASR systems with varying hyper-parameters using Recognizer Output Voting Error Reduction (ROVER) with minimal word-level edit-distance alignment (Fiscus, 1997).

MT

To train MT models for our cascaded system, we use the Fairseq (Ott et al., 2019) framework to train transformers encoder-decoder models (Vaswani et al., 2). To mitigate the exposure bias of training with ground-truth data and using ASR outputs at test time, we introduce *ASR mixing*, where during training, for each sample in the training set, the model maximizes the log-likehood of translation from both the *ground-truth source* and the *ASR source* from an ASR system. This is possible because we have triplet data for training set as well. We use the same system used in the cascaded system to generate ASR outputs for the training set. We ensemble multiple MT systems with varying random seeds using posterior combination of hypotheses during beam search.

We also train an MT model using the ESPnet toolkit (Watanabe et al., 2018) as an auxiliary model used for posterior combinations with our E2E ST systems as described in §9.3. These models use BPE vocabulary sizes that are optimal for E2E ST, which we found empirically to be smaller than for MT.

Direct Dialectal Transfer

To leverage MSA annotated speech data to improve our ASR system, we select a subset of the MGB2 data as an augmentation set to be added to the IWSLT22-Dialect data. We first use an ASR model trained on IWSLT22-Dialect data only to compute the cross-entropy of the utterances in the MGB2 data. We then select a percentage of the MGB2 utterances with the lowest cross-entropy. Similar cross-entropy based data selection has shown to effectively reduce noise resulting from domain mismatches in language modeling (Moore and Lewis, 2010) and MT (Junczys-Dowmunt, 2018). After pre-training on the mixture of MGB2 and IWSLT22-Dialect data, we then fine-tune on IWSLT22-Dialect data only.

To leverage the MSA translation data to improve our MT system, we use the OPUS corpus, cleaning sentences longer than 200 subwords. This results in about 30M sentence pairs of training data for MSA-English. We then train a larger transformer for 20 epochs on this training data. We the use fine-tune this model on the IWSLT22-Dialect data.

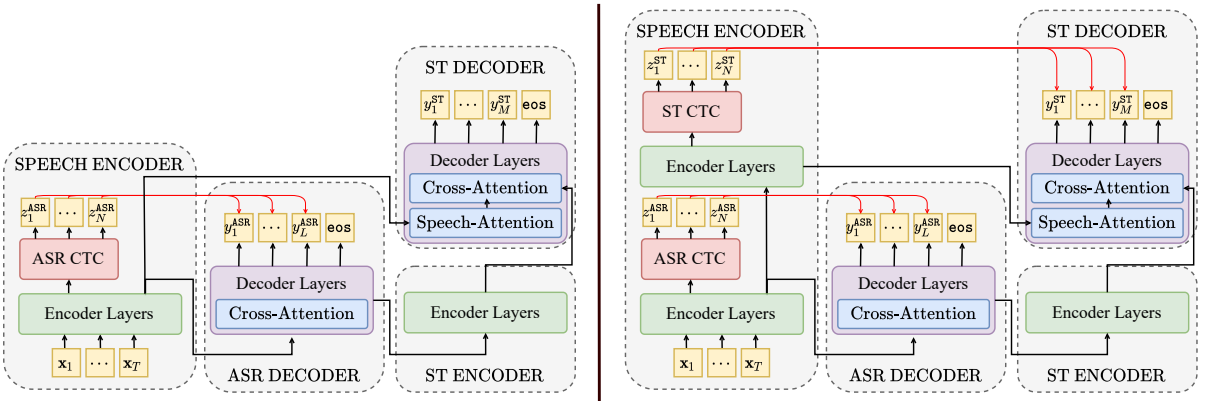

Figure 9.1: The left side presents the original Multi-Decoder architecture with searchable hidden intermediates produced by the *ASR Decoder*. The red lines indicate joint CTC/Attention decoding of beam search hypotheses produced by an autoregressive decoder. The right side presents a modified Multi-Decoder with both a hierarchical ASR to ST *Speech Encoder* optimized via CTC objectives and joint CTC/Attention ST inference.

E2E ST Systems

Multi-Decoder Architecture

Multi-decoder model (Dalmia et al., 2021b) is an end-to-end sequence model that exploits decomposition of a complex task into simpler tasks in it's model design. For speech translation it decomposes the task into ASR and MT sub-nets while maintaining the end-to-end differentiability. To train Multi-Decoder models, we modified the ESPnet framework (Watanabe et al., 2018).

As shown in figure 9.1.a, the speech signal, $X = \{\mathbf{x}_t \in \mathbb{R}^D | t = 1, ..., T\}$, is mapped to encoder representations by the *Speech Encoder* which are then in turn mapped autoregressively to decoder representations corresponding to the source language transcription, $Y^{\text{ASR}} = \{y_l^{\text{ASR}} \in \mathcal{V} | l = 1, ..., L\}$, by the *ASR Decoder*. These *ASR Decoder* representations, referred to as searchable hidden intermediates, are passed to the downstream *ST Encoder-Decoder*. In order to avoid error-propagation, the *ST Decoder* performs cross-attention over both the *Speech Encoder* and *ST Encoder* representations. The network is optimized with multi-tasking on cross-entropy losses for both the source and target languages, $\mathcal{L}_{\text{CE}}^{\text{ASR}}$ and $\mathcal{L}_{\text{CE}}^{\text{ST}}$ respectively, along with a CTC (Graves, 2012b) loss $\mathcal{L}_{\text{CTC}}^{\text{ASR}}$:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{CE}}^{\text{ASR}} + \lambda_2 \mathcal{L}_{\text{CTC}}^{\text{ASR}} + \lambda_3 \mathcal{L}_{\text{CE}}^{\text{ST}} \tag{9.1}$$

where λ's are used for interpolation. During inference, the CTC branch of the *Speech Encoder* is

also used to re-score beam search hypotheses produced by the *ASR Decoder*, following the Hybrid CTC/Attention method (Watanabe et al., 2017b).

Inaguma et al. (2021a) showed that sampling CTC output instead of always using ground truth previous token helps the Multi-Decoder model. With a CTC sampling rate of 0.2, which means that with a probability of 0.2 we would use the CTC output instead of the ground truth during training. This simulates the inference condition where there would be ASR errors. We found this technique to be particularly helpful for this dataset.

SeqKD Dialectal Transfer

Our Multi-Decoder training objective, equation 9.1, assumes that each speech signal is annotated with both a source language transcription and target language translation. In order to include additional paired MSA data into this training regime, we first generate artificial speech, transcript, and translation triplets. To do so, we first build a MSA MT model using the OPUS data. We then generate pseudo-translations for the paired MGB2 data by feeding the MSA transcriptions as inputs to the MT model. This method is based on SeqKD Kim and Rush (2016a) and can be considered as a dialectal application of MT to ST knowledge-distillation. We mix a percentage of the pseudo-translated data using the same cross-entropy based methodology as decribed in §9.3 with the Tunisian-Arabic data during training. We refer to this data augmentation as *MT SeqKD* in future sections.

Hierarchical Speech Encoder

CTC loss is often used as auxiliary loss in attention based encoder decoder models (Watanabe et al., 2017b). It helps the attention based decoder by inducing monotonic alignment with the encoder representations (Kim et al., 2017). In this work, we extend this idea by creating a hierarchical encoder that customizes the ordering of the encoder for the individual sub-tasks by using auxiliary CTC loss at each sub-task. Here, we use an auxiliary CTC loss with ASR targets and another CTC loss with ST targets. As shown in figure 9.1.b, the first 12 layers of the *Speech Encoder* produce ASR CTC alignments, $Z^{\text{ASR}} = \{z_n^{\text{ASR}} \in \mathcal{V} \cup \{\varnothing\} | n = 1, ..., N\}$, while the final 6 layers produce ST CTC alignments, $Z^{\text{ST}} = \{z_n^{\text{ST}} \in \mathcal{V} \cup \{\varnothing\} | n = 1, ..., N\}$, where $\cup\{\varnothing\}$ denotes the blank emission. This creates a hierarchical encoder structure similar to (Sanabria and Metze, 2018; Lee and Watanabe, 2021; Higuchi et al., 2022). The Multi-Decoder with hierarchical encoder is optimized with an additional ST CTC loss, $\mathcal{L}_{\text{CTC}}^{\text{ST}}$:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{CE}}^{\text{ASR}} + \lambda_2 \mathcal{L}_{\text{CTC}}^{\text{ASR}} + \lambda_3 \mathcal{L}_{\text{CE}}^{\text{ST}} + \lambda_4 \mathcal{L}_{\text{CTC}}^{\text{ST}} \tag{9.2}$$

Note that the *ST Decoder* now performs cross-attention *Speech Encoder* representations that are oriented towards the target language.

Joint CTC/Attention Decoding for ST

The ST CTC branch of the *Speech Encoder* introduced in the previous section allows us to apply joint CTC/Attention decoding using the one-pass beam search algorithm (Watanabe et al., 2017b) during ST inference as well. Although previously only applied to ASR decoding, we found that joint CTC/Attention inference for the *ST Decoder* beam search hypotheses were beneficial in this task. Deng et al. (2022a) show that joint modeling of CTC/Attention is effective for short contexts of blockwise streaming ST; as far as we know, our work is the first to show the benefit on long context. Our conjecture is that speech to translation transduction with attention mechanisms, as in the original Multi-Decoder, contains irregular alignments between the acoustic information and the target sequence. The hierarchical encoder and joint CTC/Attention decoding methods may alleviate these irregularities by enforcing greater monotonicity. We refer to the Multi-Decoder with hierarchical encoder and joint CTC/Attentionn ST decoding as the *Hybrid Multi-Decoder* in future sections.

Integrating E2E and Cascaded Systems

Guiding Multi-Decoder Representations

Since the Multi-Decoder (Dalmia et al., 2021b) uses hidden representations from the autoregressive *ASR Decoder*, we can perform search and retrieval over this intermediate stage of the model. Dalmia et al. (2021b) showed that ST quality improves by using beam search and external models like LMs to improve the representations the ASR sub-task level. We believe this an important property to have when building models for complex sequence tasks like speech translation, as often there is additional data present for the sub-tasks like ASR and MT. In this work, we help guide our Multi-Decoder model to retrieve better decoder representations by using external ASR and MT models.

We experimented with two approaches: 1) posterior level guidance and 2) surface level guidance. The former is similar in concept to posterior combination for model ensembling during inference as described in (Inaguma et al., 2020), however the Multi-Decoder allows us to incorporate both an external ASR and MT model due to the searchable hidden intermediates whereas a vanilla encoder-decoder ST model would only be compatible with an external MT model. This method requires beam search over both ASR and MT/ST for multiple models. Alternatively, surface level guidance can avoid this expensive search over the ASR intermediates by instead retrieving the hidden representations for an ASR surface sequence produced externally.

We use the ROVER ASR outputs described in §9.3 as surface level guides for the Multi-Decoder's ASR intermediates and found this to be more effective than posterior combination with external ASR models. We refer to this method of retrieval as *ROVER intermediates* in future sections. Since ROVER is based on minimal edit-distance alignment, we did not find it compatible with translation sequences. For the *ST Decoder*, we use posterior combination with external ST and MT models and refer to this as *ST/MT Posterior Combination* in future sections.

Minimum Bayes-Risk

Rather than finding the most likely translation, Minimum Bayes-Risk (MBR) decoding aims to find the translation that maximizes the expected *utility* (equivalently, that minimizes *risk*, (Kumar and Byrne, 2002, 2004; Eikema and Aziz, 2020)). Let $\bar{\mathcal{Y}}_{\text{cands}}$, $\bar{\mathcal{Y}}_{\text{samples}}$ be sets containing N *candidate* hypotheses and M *sample* hypothesis. This sets can be obtained from one or multiple model by, for example sampling or taking the top beams in beam search. Let $u(y^*, y)$ be an utility function measuring the similarity between a hypothesis y and a reference y (we only consider BLEU in this work). MBR decoding seeks for

$$\hat{y}_{\text{MBR}} = \underset{y \in \bar{\mathcal{Y}}_{\text{cands}}}{\text{argmax}} \underbrace{\mathbb{E}_{Y \sim p_\theta(y|x)}[u(Y, y)]}_{\approx \frac{1}{M} \sum_{j=1}^{M} u(y^{(j)}, y)} , \tag{9.3}$$

We experimented with using MBR as a technique for system combination, in two forms:

- *True*: the stronger system (the E2E) is used to generate the N candidates $\bar{\mathcal{Y}}_{\text{cands}}$ and the weaker system (the Cascaded system) is used to generate M samples $\bar{\mathcal{Y}}_{\text{samples}}$. This means that the outputs will guaranteed to generated by the E2E system.

- *Joint*: in this case, both the E2E and the Cascaded generate N hypotheses, with are then concatenated to make both the candidate set and sample set $\bar{\mathcal{Y}}_{\text{samples}} = \bar{\mathcal{Y}}_{\text{cands}}$, with $|\bar{\mathcal{Y}}_{\text{cands}}| = 2N$

We explored using beam search and nucleus sampling (Holtzman et al., 2019) with different p values for both generating candidates and generating samples to compute the expectation over. Overall we found that, for both settings, using beam search to generate hypothesis for the E2E model and nucleus sampling with $p = 0.9$ for the cascaded system yield the best results. We use $N = M = 50$ for both settings.

9.4 Experimental Setup

ASR: We extracted 80-channel log-mel filterbank coefficients computed with 25-ms window size and shifted every 7-ms with 3-dimensional pitch features.[4] The features were normalized by the mean and the standard deviation calculated on the entire training set. We applied SpecAugment (Park et al., 2019) with mask parameters $(m_T, m_F, T, F) = (5, 2, 27, 0.5)$ and bi-cubic time-warping. We use a BPE vocabulary size of 1000. Our encoder has 2 CNN blocks followed by 12 Conformer blocks following (Guo et al., 2021). Each CNN block consisted of a channel size of 256 and a kernel size of 3 with a stride of 2×2, which resulted in time reduction by a factor of 4. Our decoder has 6 Transformer blocks. In both encoder and decoder blocks, the dimensions of the self-attention layer d_{model} and feed-forward network d_{ff} were set to 256 and 2048, respectively. The number of attention heads H was set to 8. The kernel size of depthwise separable convolution in Conformer blocks was set to 31. We optimized the model with the joint CTC/attention objective with a CTC weight of 0.3. We also used CTC and LM scores during decoding. Models were trained for 60 epochs. We averaged the model parameters of the 10 best epoch checkpoints by validation loss. Our LM is a BLSTM with 4 layers and 2048 unit dimension. Beam search is performed with beam size 20, CTC weight 0.2, and LM weight 0.1.

MT: We use SentencePiece (Kudo and Richardson, 2018b) with the Byte-pair Encoding algorithm (Sennrich et al., 2016). We experimented with various vocabularies sizes and found that 4000 vocabulary size to be the best for small models. For the pretrained model, we use a vocabulary size of 16000. The small transformer model used for the non-dialect submissions has 512 embedding dimensions, 1024 feedforward dimensions, 6 layers and 4 heads on each layer on both encoder/decoder. The large transformer model used for dialect transfer has 1024 embedding dimensions, 4096 feedforward dimensions, 6 layers and 16 heads on each layer on both encoder/decoder. Models were trained with early stopping by validation loss. We averaged the model parameters of the last 5 epoch checkpoints. Unless otherwise specified, we use beam search with beam size of 5 and no length penalty in beam search.

Multi-Decoder: We use the same feature extraction as for ASR. We use separate BPE vocabularies for source and target, both of size 1000. The ASR sub-net of the Multi-Decoder is also the same as our ASR configuration, allowing for pre-trained initialization of the ASR encoder, decoder, and CTC. The hierarchical encoder adds 6 additional Transformer layers to the original 12 Conformer layers. The MT sub-net of the Multi-Decoder has a 2 layer Transformer encoder and a 6 layer Transformer decoder. This second encoder has no convolutional subsampling. The MT sub-net has the same d_{model} and d_{ff} as the ASR sub-net. We optimized the model a CTC weight

[4] 7-ms shift was found to be helpful due to the presence of many short utterances in the IWSLT22-Dialect data.

ID	Model Type / Name	Dialect Transfer	test1 WER(↓)
A1	ASR Conformer	✗	50.4
A2	+ ROVER Comb.	✗	48.1
A3	ASR Conformer	✓	50.0
A4	+ ROVER Comb.	✓	**47.5**
			MT BLEU(↑)
B1	MT Transformer (Fairseq)	✗	21.8
B2	+ Posterior Comb.	✗	22.8
B3	MT Transformer (Fairseq)	✓	22.4
B4	+ Posterior Comb.	✓	**23.6**
B5	MT Transformer (ESPnet)	✗	21.0

Table 9.2: Results of the ASR and MT components of our cascaded systems, as measured by % WER and BLEU score on the provided test1 set. ROVER and posterior combinations were applied to ASR and MT respectively.

of 0.3 and an ASR weight of 0.3. Models were trained for 40 epochs. We averaged the model parameters of the 10 best epoch checkpoints by validation loss. Beam search over the ASR-subnet uses the same setting as for ASR. Beam search over the MT-subnet uses beam size 5/10 with CTC weight 0.3/0.1 for the basic/dialect conditions. Length penalty 0.1 was used for all cases.

9.5 Results and Analyses

Submitted Shared Task Systems

Figure 9.2 shows the results for ASR and MT systems used as part of the cascaded system as evaluated by WER and BLEU score respectively on the provided test set, test1. Dialectal transfer provides a moderate boosts of 0.4% and 0.6% WER without ROVER and with ROVER respectively. Notably, WER's for all systems are relatively high despite a moderate amount of training data; this is perhaps due to the non-standard orthographic form of the Tunisian-Arabic transcriptions.[5] Another possible cause for the high WER is the conversational nature of the data, which may require normalization similar to the Switchboard dataset (Godfrey et al., 1992). For the MT systems, we see that posterior combination leads to over 1 BLEU point improvements when trans-

[5]We found that the WER's decreased by about 4% when removing diacritics from the hypothesis and the reference.

ID	Type	Model Name	Child System(s)	Dialect Transfer	test1 BLEU(↑)	test2 BLEU(↑)
C1	Cascade	ASR Mixing Cascade	A1,B1	✗	16.4	-
C2	Cascade	+ ASR Rover Comb.	A2,B1	✗	16.7	-
C3	Cascade	+ MT Posterior Comb.	A2,B2	✗	17.5	18.6
C4	Cascade	ASR Mixing Cascade	A3,B3	✓	17.3	-
C5	Cascade	+ ASR Rover Comb.	A4,B3	✓	17.4	-
C6	Cascade	+ MT Posterior Comb.	A4,B4	✓	**17.9**	**19.4**
D1	E2E ST	Hybrid Multi-Decoder	-	✗	17.7	-
D2	Mix	+ ROVER Intermediates	A2	✗	18.1	19.1
D3	Mix	+ ST/MT Posterior Comb.	A2,B5	✗	18.7	19.7
D4	E2E ST	Hybrid Multi-Decoder	-	✓	18.2	-
D5	Mix	+ ROVER Intermediates	A4	✓	18.3	19.5
D6	Mix	+ ST/MT Posterior Comb.	A4,B5	✓	**18.9**	**19.8**
E1	Mix	Min. Bayes-Risk Ensemble	C3,D3	✗	19.2	20.4
E2	Mix	Min. Bayes-Risk Ensemble	C6,D6	✓	**19.5**	**20.8**

Table 9.3: Results of our cascaded, E2E, and integrated cascaded/E2E systems as measured by BLEU score on the blind test2 and provided test1 sets. *Dialect Transfer* indicates the use of either MGB2 or OPUS data. Rover, posterior combinations, and minimum bayes-risk ensembling were applied to both cascaded and E2E systems, with *Child System(s)* indicating the inputs to the resultant systems combinations.

lating ground-truth source sentences. Interestingly, while there is some benefit from the dialectic transfer, the benefits are relatively small, yielding an additional 0.8 BLEU for the ensembled models. This might be due to the domain mismatch between the Tunisian-Arabic data and MSA data.

Figure 9.3 shows the results of our cascaded, E2E, and integrated cascaded/E2E systems on both the blind shared task test set, test2, and on the provided test set, test1. The *Hybrid Multi-Decoder* outperforms the *ASR Mixing Cascade* by 1.3 and 0.9 BLEU on test1 without and with dialectal transfer respectively. Both models are boosted by the use of ROVER. The benefit of ROVER for models without dialectal transfer (0.3 BLEU) was larger than for models with dialectal transfer (0.1 BLEU), showing some diminishing returns from isolated improvements of the ASR component of the overall ST task. Posterior combination provided boosts in the range of 0.5-0.8 BLEU across the models. Finally, the *Minimum Bayes Risk Ensembling* yielded additional gains of 0.6-1.3 BLEU. The differences between the final *Minimum Bayes Risk Ensembling* systems and the best single systems without any external model integration are 1.5 and 1.3 BLEU without and

Task	MGB2 Training Data	test1 WER(↓)
ASR	none	53.1
ASR	8% w/ random select	52.7
ASR	8% w/ CE filter	**52.4**
ASR	25% w/ CE filter	**52.4**
ASR	50% w/ CE filter	53.0
ASR	75% w/ CE filter	53.5
		BLEU(↑)
ST	none	16.6
ST	25% w/ CE filter + MT SeqKD	**17.1**
ST	50% w/ CE filter + MT SeqKD	**17.1**

Table 9.4: Ablation study on the effects of additional MGB2 data on ASR and ST performance as measured by WER and BLEU on the test1 set respectively.

Model Name	test1 ST BLEU(↑)	test1 MT BLEU(↑)
MT Transformer	16.2	20.9
+ ASR Mixing Training	**16.7**	**21.8**

Table 9.5: Ablation study on the effects of ASR mixing on ST and MT as measured by BLEU on the test1 set.

without dialectal transfer respectively.

Ablation Studies

To show the individual contributions of our various methods, we present in this section several ablations. First, we show in figure 9.4 the impact of dialectal transfer from MGB2 data on ASR (as described in §9.3) and on E2E ST (as described in §9.3). As subset of MGB2 data selected via the cross-entropy filter outperformed a randomly selected subset, although both were better than when no MGB2 data was included. Since the IWSLT22-Dialect utterances were shorter than the MGB2 utterances on average, one effect of the cross-entropy filter was the removal of long utterances

Model Name	test1 BLEU(↑)
Encoder-Decoder	16.0
Multi-Decoder	17.1
+ ASR CTC Sampling	17.6
+ Hierarchical Encoder	17.9
+ Joint CTC/Attn ST Decoding (D4)	18.2
+ ASR CTC Sampling	**18.4**

Table 9.6: Ablation study on the effects of ASR CTC sampling, hierarchical encoder, and joint CTC/Attn ST decoding as measured by BLEU on the test1 set.

Model Name	MBR Method	Dialect Transfer	test1 BLEU(↑)	test2 BLEU(↑)
MBR Ensemble	True	✗	19.0	20.1
MBR Ensemble (E1)	Joint	✗	**19.2**	**20.4**
MBR Ensemble	True	✓	19.3	20.7
MBR Ensemble (E2)	Joint	✓	**19.5**	**20.8**

Table 9.7: Comparison of the true vs. joint methods for minimum bayes-risk ensembling as measured by BLEU on the test1 and test2 sets.

Model Name	test2 BLEU(↑)	test2 DA Ave. / z-score(↑)
Hybrid Multi-Decoder (D6)	19.8	66.5 / 0.119
MBR Ensemble (E2)	**20.8**	66.5 / 0.114

Table 9.8: Human evaluation results, as measured by DA average and z-score, showing the impact of maximizing BLEU score via minimum bayes-risk ensembling.

which appeared to benefit the model. We found that using up to 25% of the MGB2 data was best for ASR. For ST, both 25% and 50% of the MGB2 data with *MT SeqKD* yielded 0.5 BLEU gains, which is slightly less than the 0.8 BLEU gains that our cascaded systems obtained from dialectal

transfer. This suggests some that there our *MT SeqKD* method may be improved in the future.

Next, in figure 9.5 we show the results MT and ST systems trained with and without *ASR mixing* (as described in §9.3), both in the cascaded setting and using ground-truth source sentences. Overall we see that *ASR mixing* helps improving the cascaded system. Surprisingly this also improves results for the translating from ground-truth source sentences. We hypothesise that *ASR mixing* acts as a form of regularization for the orthographic inconsistencies in the source transcriptions due to the conversational nature of Tunisian-Arabic.

In table 9.6, we show the effects of the *ASR CTC Sampling*, *Hierarchical Encoder*, and *Joint CTC/Attention ST Decoding* modifications to the original Multi-Decoder (as described in §9.3). We found that each of these techniques boosts the overall performance and we also found their effects to be additive. Table 9.6 also shows the performance of a vanilla encoder-decoder for comparison, which performed significantly worse than the Multi-Decoder. Due to time limitations, we did not submit the Multi-Decoder with hierarchical encoder, joint CTC/Attention ST decoding, and ASR CTC sampling for shared task evaluation, but this was our strongest single system as evaluated on the test1 set.

Finally, Figure 9.7 shows the results for the two different settings for system combination through MBR (as described in §9.3). Using the *Joint* setting where the hypothesis from both system are considered as both candidates/samples leads to the best translations compared to the *True* setting. Figure 9.8 shows that while effective for maximizing BLEU score, MBR did not improve according to human evaluation.[6]

9.6 Conclusion

In this chapter, we have presented CMU's dialect speech translation systems for IWSLT 2022. Our systems encompass various techniques across cascaded and E2E approaches. Of the techniques we presented, the hierarchical encoder and joint CTC/Attention ST decoding modifications to the Multi-Decoder and the minimum bayes-risk ensembling were amongst the most impactful. In future work, we seek to formalize these methods with additional theoretical and experimental backing, including extensions to other corpora and tasks such as pure MT.

We've now established the efficacy of sequential, non-temporal multi-sequence modeling for offline speech translation. Next, we consider how to introduce the temporal element, as required for streaming speech translation.

[6]Human evaluation methodology is detailed in (Anastasopoulos et al., 2022)

Part IV

Sequential, Temporal Multi-Sequence Modeling

Chapter 10

Hierarchical CTC: Improving Auto-regressive Translation with Temporal Alignment

Summary

This chapter delves into **sequential, temporal** multi-sequence modeling. We start by proving out the approach in an offline setting to demonstrate the general efficacy; then Chapter 9 evaluates the approach under a streaming setting.

Connectionist Temporal Classification (CTC) is a widely used approach for automatic speech recognition (ASR) that performs conditionally independent monotonic alignment. However for translation, CTC exhibits clear limitations due to the contextual and non-monotonic nature of the task and thus lags behind attentional decoder approaches in terms of translation quality. In this work, we argue that CTC *does* in fact make sense for translation if applied in a joint CTC/attention framework wherein CTC's core properties can counteract several key weaknesses of pure-attention models during training and decoding. To validate this conjecture, we modify the Hybrid CTC/Attention model originally proposed for ASR to support text-to-text translation (MT) and speech-to-text translation (ST). Our proposed joint CTC/attention models outperform pure-attention baselines across six benchmark translation tasks.

10.1 Introduction

Automatic speech recognition (ASR), machine translation (MT), and speech translation (ST) have conspicuous differences but are all closely related sequence-to-sequence problems. Researchers

from these respective fields have long recognized the opportunity for cross-pollinating ideas (He and Deng, 2011), starting from the coupling of statistical ASR (Huang et al., 2014) and MT (Al-Onaizan et al., 1999) which gave rise to early approaches for ST (Waibel, 1996; Ney, 1999). Notably in the end-to-end era, attentional encoder-decoder approaches emerged in both MT (Bahdanau et al., 2015) and ASR (Chorowski et al., 2015; Chan et al., 2015), rising to great prominence in both fields.

During this same period, there has been another prominent end-to-end approach in ASR: Connectionist Temporal Classification (CTC) (Graves et al., 2006). Unlike the highly flexible attention mechanism which can handle ASR, MT, and ST alike, CTC models sequence transduction as a monotonic alignment of inputs to outputs and thus fits more naturally with ASR than it does with translation. Still, many interested in non-autoregressive translation have applied CTC to MT (Libovický and Helcl, 2018) and ST (Inaguma et al., 2021b) and promising techniques have emerged, shrinking the gap between autoregressive approaches (Saharia et al., 2020; Gu and Kong, 2021; Chuang et al., 2021; Huang et al., 2022). These recent developments suggest that the latent alignment ability of CTC is a promising direction for translation – this leads us to question: *can CTC alignments improve autoregressive translation?* In particular, we are interested in frameworks that leverage the strength of CTC while minimizing its several harmful incompatibilities (see §10.3) with translation tasks.

Inspired by the success of Hybrid CTC/Attention in ASR (Watanabe et al., 2017b), we investigate jointly modeling CTC with an autoregressive attentional encoder-decoder for translation. Our conjecture is that the monotonic alignment and conditional independence of CTC, which weaken purely CTC-based translation, counteract particular weaknesses of attentional models in joint CTC/attention frameworks. In this work, we seek to investigate *how* each CTC property interacts with corresponding properties of the attentional counterpart during joint training and decoding. We design a joint CTC/attention architecture for translation (§10.4) and then examine the positive interactions which ultimately result in improved translation quality compared to pure-attention baselines, as demonstrated on the IWSLT (Cettolo et al., 2012), MuST-C (Di Gangi et al., 2019), and MTedX (Salesky et al., 2021) MT/ST corpora (§10.6).[1]

10.2 Background: Joint CTC/Attn for ASR

Both the CTC (Graves et al., 2006) and attentional encoder-decoder (Bahdanau et al., 2015) frameworks seek to model the Bayesian decision seeking the output, $\hat{Y}$, from all possible sequences, $\mathcal{V}^{\text{tgt}}*$, by selecting the sequence which maximizes the posterior likelihood $P(Y|X)$,

[1]Models are are available in ESPnet. For ST, refer to `egs2/must_c_v2/st1` and for MT refer to `egs2/iwslt14/mt1`.

CTC	ATTENTION	JOINT CTC/ATTENTION	ASR	MT/ST
$P_{\text{CTC}}(Y\|X) \triangleq \sum_{Z \in \mathcal{Z}} \prod_{t=1}^{T} P(z_t\|X, \cancel{z_{1:t-1}})$	$P_{\text{Attn}}(Y\|X) \triangleq \prod_{l=1}^{L} P(y_l\|y_{1:l-1}, X)$	$P_{\text{Joint}}(Y\|X) \triangleq P_{\text{CTC}}(Y\|X)^{\lambda} \times P_{\text{Attn}}(Y\|X)^{1-\lambda}$	✓	✓
Hard Alignment Criterion only allows monotonic alignments of inputs to outputs	**Soft Alignment** Flexible attention-based input-to-output mappings may overfit to irregular patterns	**During Training:** Hard alignment objective produces stable encoder representations allowing the decoder to more rapidly learn soft alignment patterns	✓	**L1** See §10.3
Conditional Independence Assumes that there are no dependencies between each output unit given the input	**Conditional Dependence** Locally normalized models with output dependency exhibit label/exposure biases	**During Decoding:** Use of conditionally independent likelihoods in joint scoring eases the exposure/label biases from conditionally dependent likelihoods	✓	**L2** See §10.3
Input-Synchronous Emission Each input representation emits exactly one blank or non-blank output token	**Autoregressive Generation** Need to detect end-points and compare hypotheses of different length in beam search	**During Decoding:** Input-synchronous emission determines output length based on input length counteracting the autoregressive end-detection problem	✓	**L3** See §10.3

Table 10.1: Description of three reasons why joint CTC/attention modeling is powerful in ASR. In order to understand whether these positive interactions between properties of the CTC and attention frameworks are applicable to MT/ST, we must address three corresponding concerns, L1-3, about the applicability of CTC to translation (§10.2).

where $X = \{\mathbf{x}_t \in \mathcal{S}^{\text{src}} | t = 1, ..., T\}$ and $Y = \{y_l \in \mathcal{V}^{\text{tgt}} | l = 1, ..., L\}$. The source set $\mathcal{S}^{\text{src}}$ is a discrete vocabulary in the MT case and a continuous real space in the ST case while the target set $\mathcal{V}^{\text{tgt}}$ is always a discrete vocabulary. Note that the T-length of the input is assumed to be longer than the L-length output for speech tasks (Graves et al., 2006), but this is not necessarily true for MT.

What are the critical differences between the CTC and attention frameworks? As shown in the first two columns of Table 10.1, CTC and attention offer different formulations of the posterior likelihood, $P_{\text{CTC}}(\cdot)$ and $P_{\text{Attn}}(\cdot)$ respectively. First of all, the attention mechanism is a flexible input-to-output mapping function which allows a decoder to perform **soft alignment** of an output unit y_l to multiple input units $\mathbf{x}_{[...]}$ without restriction. One downside of this flexibility is a risk of destabilized optimization (Kim et al., 2017). CTC on the other hand marginalizes the likelihoods of all possible input to alignment sequence, $Z = \{z_t \in \mathcal{V}^{\text{tgt}} \cup \{\varnothing\} | t = 1 \dots T\}$, mappings via **hard alignment** where each output unit z_t maps to a single input unit $\mathbf{x}_t$ in a strictly monotonic pattern. $\varnothing$ is a "blank" and Z maps deterministically to Y by removing blanks and repeated emissions.

Secondly, the attentional decoder models each output unit y_1 with **conditional dependence** on not only the input X, but also the previous output units $y_{1:l-1}$. In contrast, CTC makes a **conditional independence** assumption that each z_t does not depend on $z_{1:t-1}$ if already conditioned on X (as denoted by the strike-through in Table 10.1) – this is a strong assumption which allows for efficient computation of marginalized likelihoods over all $Z \in \mathcal{Z}(Y, T)$ via dynamic programming. On the plus, since CTC does not model causality between output units it is not plagued by

the same label and exposure biases that exist in attentional decoders due to local normalization of causal likelihoods (Bottou, 1991; Ranzato et al., 2016; Hannun, 2019).

Finally, the attentional decoder is an **autoregressive generator** that decodes the output until a stop token, <eos>. Comparing likelihoods for sequences of different lengths requires a heuristic brevity penalty. Furthermore label bias with respect to the stop token manifests as a length problem where likelihoods degenerate for unexpectedly long outputs (Murray and Chiang, 2018b). In comparison, CTC is an **input-synchronous emitter** that consumes an input unit in order to produce an output unit. Therefore, CTC cannot produce an output longer than the input representation which feeds the final posterior output layer – but this also means that CTC does not require end detection.

As previously shown by (Kim et al., 2017; Watanabe et al., 2017b), jointly modeling CTC and an attentional decoder is highly effective in ASR. The foundation of this architecture is a shared encoder, ENC, which feeds into both CTC, $P_{\text{CTC}}(\cdot)$, and attentional decoder, $P_{\text{Attn}}(\cdot)$, posteriors:

$$\mathbf{h} = \text{Enc}(X) \tag{10.1}$$

$$P_{\text{CTC}}(z_t|X) = \text{CTC}(\mathbf{h}_t) \tag{10.2}$$

$$P_{\text{Attn}}(y_l|X, y_{1:l-1}) = \text{Dec}(\mathbf{h}, y_{1:l-1}) \tag{10.3}$$

where $\text{CTC}(\cdot)$ denotes a linear projection to the CTC output vocabulary, $\mathcal{V}^{\text{tgt}} \cup \{\varnothing\}$, followed by softmax. $\text{DEC}(\cdot)$ denotes autoregressive decoder layers followed by a linear projection to the decoder output vocabulary, $\mathcal{V}^{\text{tgt}} \cup \{\text{<eos>}\}$, and softmax. The joint network is optimized via a multi-tasked objective, $\mathcal{L}^{\text{ASR}} = \mathcal{L}^{\text{ASR}}_{\text{CTC}} + \lambda\mathcal{L}^{\text{ASR}}_{\text{Attn}}$, where λ interpolates the CTC and decoder cross-entropy losses.

Joint decoding is typically performed with a one-pass beam search where CTC plays a secondary role as a joint scorer while attention leads the major hypothesis expansion and end detection functions in the algorithm (Watanabe et al., 2017b; Tsunoo et al., 2021). However, CTC is capable of taking over the lead role if called upon (e.g. for streaming applications) (Moritz et al., 2019).

10.3 Potential CTC Limitations in MT/ST

Why exactly does this joint CTC/attention framework perform so well in ASR? As summarized in column 3 of Table 10.1, we are particularly interested in three reasons which arise from the combination of the hard vs. soft alignment, conditional independence vs. dependence, and input-synchronous emission vs. autoregressive generation properties of CTC and attention respectively. These dynamics have become well understood in ASR, owing to the popularity of the joint framework (Watanabe et al., 2018) amongst ASR practitioners.

So can CTC and attention also complement each other when applied jointly to translation?[2] ASR, MT, and ST can all be generalized as sequence transduction tasks following the Bayesian formulation. Attentional decoders have been a predominant technical solution to each of these tasks. However, the CTC framework appears to have several limitations specific to MT/ST that are not present in ASR; this seemingly diminishes the promise of the joint CTC/attention framework for translation. In this work, we seek to address the following three concerns about MT/ST CTC which appear to inhibit the CTC/attention framework (please refer back to Table 10.1 as needed).

L1 *Can CTC encoders perform sophisticated input-to-output mappings required for translation?*

Unlike ASR, translation entails non-monotonic mappings due to variable word-ordering across languages. Additionally, inputs may be shorter than outputs as mappings are not necessarily one-to-one. Furthermore, the mapping task for ST is compositional where logically a speech signal first maps to a source language transcription before being mapped to the ultimate translation. All of these complications appear to directly contradict the **hard alignment** of CTC. If CTC cannot produce stable encoder representations for MT/ST, then during joint training attention does not receive the optimization benefit as in ASR (per row 2 of Table 10.1). Fortunately, prior works suggest that these challenges are not insurmountable. Chuang et al. (2021) showed that self-attentional encoders can perform latent model variable word orders for ST, Libovický and Helcl (2018); Dalmia et al. (2022) proposed up-sampling encoders that produce expanded input representations for MT, and Sanabria and Metze (2018); Higuchi et al. (2022) proposed hierarchical CTC encoders that can compose multiple output resolutions for ASR. In §10.4, we incorporate these techniques into a unified hierarchical CTC encoding method for MT/ST which is capable of sophisticated input-to-output mappings.

L2 *Does CTC-based translation quality lag too far behind attention-based to be useful?*

CTC-based ASR has recently shown competitive performance due in large part to improved neural architectures (Gulati et al., 2020) and self-supervised learning (Baevski et al., 2020; Hsu et al., 2021), but the gap between CTC and attention for translation appears to be greater (Gu and Kong, 2021). Perhaps the **conditional independence** of CTC inhibits the quality to such a degree in MT/ST where these likelihoods cannot ease the label/exposure biases of the attentional decoder as they do in ASR (per row 3 of Table 10.1). The relative weakness of non-autoregressive translation approaches has been well-studied. Knowledge distillation (Kim and Rush, 2016b; Zhou et al., 2019) and iterative methods (Qian et al., 2021; Chan et al., 2020; Huang et al., 2022) all attempt to bridge the gap between non-autoregressive models and their autoregressive counterparts. In §10.6, we address this concern empirically; we find that even CTC models with 28% relative

[2]This particular question has not been addressed in literature. For an account of related works, please see §10.8.

BLEU reduction compared to attention yield improvements when CTC and attention are jointly decoded.

L3 *Is the alignment information produced by CTC-based translation models reasonable?*

In ASR, CTC alignments are reliable enough to segment audio data by force aligning inputs to target transcription outputs (Kürzinger et al., 2020) and exhibit minimal drift compared to hidden Markov models (Sak et al., 2015). However, CTC alignments are not as well studied in translation. It is an open question of whether or not the **input-synchronous emission** of CTC for translation has sufficient alignment quality to support the end detection responsibility during joint decoding as it does in ASR (per row 4 of Table 10.1). Ideally, the CTC alignments are strong enough such that CTC can lead joint decoding by proposing candidates for hypothesis expansion in each beam step until all input units are consumed (at which point the end is detected), as in an input-synchronous beam search. More conservatively, the CTC alignments may be too unreliable to take the lead but could still guide the attentional decoder's end detection by penalizing incorrect lengths via joint scoring, as in an output-synchronous beam search. In §10.4, we lay out comparable forms for input and output-synchronous beam search which allows us to examine the impact on translation quality depending on whether CTC is explicitly responsible for or only implicitly contributing to end detection.

10.4 Joint CTC/Attention for Translation

Hierarchical CTC Encoding

Per L1 described in §10.3, we seek to build a CTC encoder for translation which handles sophisticated input-to-output mappings. Unlike ASR where outputs are assumed to be 1) always shorter than inputs and 2) monotonic with respect to inputs, translation needs to account for variability of lengths and and word orderings. We therefore propose to use a hierarchical CTC encoding scheme which 1) aligns inputs to length-adjusted *source*-oriented encodings before 2) aligning to re-ordered *target*-oriented encodings, as shown in Figure 10.1. Our encoding process thus consists of two compartmentalized functions: length-adjustment and re-ordering.

Length-adjustment For MT, we up-sample the lengths of the source-oriented encodings in order to output sequences longer than the input. For ST, we down-sample the lengths of the source-oriented encodings to coerce a discrete textual representation of the real-valued speech input. We enforce source orientations using CTC criteria that seek to align the length adjusted intermediate layer encoder representations towards source text sequences (for MT this is the same as the input

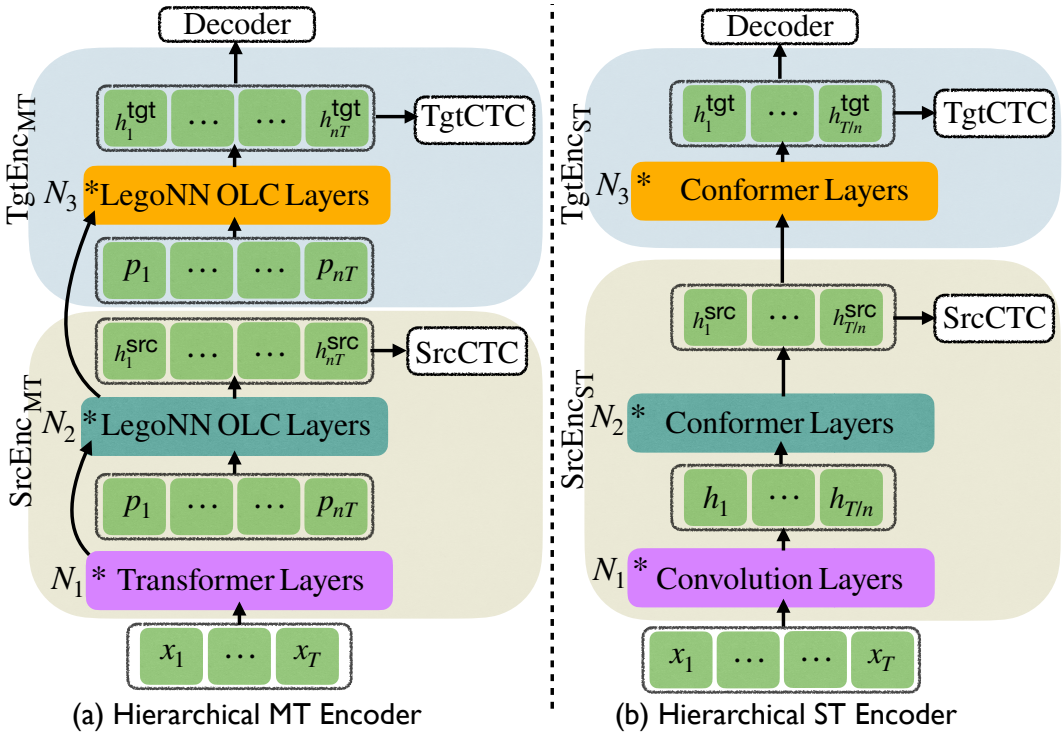

Figure 10.1: Hierarchical MT/ST encoders where representations are first up/down-sampled by $\text{SRCENC}_{\text{MT/ST}}$ and then re-ordered by $\text{TGTENC}_{\text{MT/ST}}$.

and for ST this is the ASR target). By compartmentalizing length-adjustment within this initial stage, we allow subsequent encoder layers to focus solely on re-ordering.

Re-ordering We then obtain target-oriented encodings with subsequent encoder layers, where re-ordering is enforced using CTC criteria that seek to align final layer encoder representations towards target text sequences. Critically, the underlying neural network architecture must be able to model latent re-ordering as the CTC criterion itself will only consider monotonic alignments of the final encoder representation to the target.

Our proposed MT/ST hierarchical encoders consist of the following components:

$$\mathbf{h}^{\text{SRC}} = \text{SRCENC}_{\text{MT/ST}}(X) \tag{10.4}$$

$$P_{\text{CTC}}(z_t^{\text{SRC}}|X) = \text{SRCCTC}_{\text{MT/ST}}(\mathbf{h}_t^{\text{SRC}}) \tag{10.5}$$

$$\mathbf{h}^{\text{TGT}} = \text{TGTENC}_{\text{MT/ST}}(\mathbf{h}^{\text{SRC}}) \tag{10.6}$$

$$P_{\text{CTC}}(z_t^{\text{TGT}}|X) = \text{TGTCTC}_{\text{MT/ST}}(\mathbf{h}_t^{\text{TGT}}) \tag{10.7}$$

The hierarchical encoders are jointly optimized with an attentional decoder using a multi-tasked

Algorithm 1 *Output*-Synchronous Step Function: attentional decoder proposes candidates to expand hypotheses which are all of l-length at step l.

procedure OUTPUTSTEP(prtHs, $X, l, p, \max L$)
  newPrtHs $= \{\}$; endHs $= \{\}$
  for $y_{1:l-1} \in$ prtHs **do**
    attnCnds $=$ top-k$(P_{\text{Attn}}(y_l|X, y_{1:l-1}), \text{k} = p)$
    for $c \in$ attnCnds **do**
      $y_{1:l} = y_{1:l-1} \oplus \text{c}$
      $\alpha_{\text{CTC}} = \text{CTCScore}(y_{1:l}, X_{1:T})$
      $\alpha_{\text{Attn}} = \text{AttnScore}(y_{1:l}, X_{1:T})$
      $\beta = \text{LengthPen}(y_{1:l})$
      $P_{\text{Beam}}(y_{1:l}|X) = \alpha_{\text{CTC}} + \alpha_{\text{Attn}} + \beta$
      if (c is <eos>) or (l is max$L$) **then**
        endHs$[y_{1:l}] = P_{\text{Beam}}(\cdot)$
      else
        newPrtHs$[y_{1:l}] = P_{\text{Beam}}(\cdot)$
      end if
    end for
  end for
  return newPrtHs, endHs
end procedure

Hypothesis Expansion

Joint Scoring

End Detection

Algorithm 2 *Input*-Synchronous Step Function: CTC proposes candidates to expand hypotheses which are all produced from t input units at step t.

procedure INPUTSTEP(prtHs, X, t, p, T)
  newPrtHs $= \{\}$; endHs $= \{\}$
  CTCCnds $=$ top-k$(P_{\text{CTC}}(z_t|X), \text{k} = p)$
  for $y \in$ prtHs **do**
    for $c \in$ CTCCnds **do**
      if (c is $\varnothing$) or (c is repeat) **then**
        $\tilde{y} = y$
      else
        $\tilde{y} = y \oplus \text{c}$
      end if
      $\alpha_{\text{CTC}} = \text{CTCScore}(\tilde{y}, X_{1:t})$
      $\alpha_{\text{Attn}} = \text{AttnScore}(\tilde{y}, X_{1:T})$
      $\beta = \text{LengthPen}(\tilde{y})$
      $P_{\text{Beam}}(\tilde{y}|X) = \alpha_{\text{CTC}} + \alpha_{\text{ATTN}} + \beta$
      if t is T **then**
        endHs$[\tilde{y}] = P_{\text{Beam}}(\cdot)$
      else
        newPrtHs$[\tilde{y}] = P_{\text{Beam}}(\cdot)$
      end if
    end for
  end for
  return newPrtHs, endHs
end procedure

objective, $\mathcal{L} = \mathcal{L}_{\text{SRCCTC}} + \lambda_1 \mathcal{L}_{\text{TGTCTC}} + \lambda_2 \mathcal{L}_{\text{ATTN}}$, where λ's interpolate source-oriented CTC, target-oriented CTC, and decoder cross-entropy losses.

As shown in Figure 10.1.a, $\text{SRCENC}_{\text{MT}}(\cdot)$ consists of N_1 Transformer (Vaswani et al., 2017) layers followed by N_2 up-sampling Output Length Controller (OLC) layers used in LegoNN (Dalmia et al., 2022) – the layer-wise positional embeddings of the OLC architecture enable latent length-adjustment of textual inputs. $\text{TGTENC}_{\text{MT}}(\cdot)$ consists of N_3 non-up-sampling OLC layers – the layer-wise attention of the OLC architecture enables latent re-ordering. Our ST encoder is similar, but uses Conformers (Gulati et al., 2020) to capture the local and global dependencies in speech, as shown in Figure 10.1.b. $\text{SRCENC}_{\text{ST}}(\cdot)$ consists of N_1 convolutional blocks (Dong et al., 2018) for down-sampling followed by N_2 Conformer layers – this stage is analogous to the ASR sub-task of ST where a long speech signal is length-adjusted to a shorter latent textual representation. $\text{TGTENC}_{\text{ST}}(\cdot)$ consists of N_3 Conformer layers – this stage is analogous to the MT sub-task of ST where latent re-ordering is enabled by self-attention. LegoNN and Conformer are further described in §10.10.

Input/Output-Synchronous Decoding

Per L2 and L3 described in §10.3, we seek to design a joint decoding algorithm with input and output-synchronous variants of one-pass beam search which differ only in whether CTC or attention takes the leading role. As shown in Algorithms 1 and 2, we propose to align the input and

output beam-step functions along three common functions: hypothesis expansion, joint scoring, and end detection. Using these mirrored forms, let us now interpret the respective roles of CTC and attention.

Output-Synchrony Consider first that attention is in the leading role, which means that we are working with an output-synchronous beam search. Note that this is the algorithm originally proposed by Hori et al. (2017a). OUTPUTSTEP performs *hypothesis expansion* by computing the attentional decoder's output posterior at label step l, $P_{\text{Attn}}(y_l|X, y_{1:l-1})$ for each partial hypothesis, $y_{1:l-1}$. A pre-beam size, p, is then used to select the top candidate output units (Seki et al., 2019a), $\mathrm{attnCnds}$, which are used to expand the partial hypotheses via concatenation, denoted by $\oplus$. In the *joint scoring* block, the attentional decoder likelihood, $\mathrm{AttnScore}(\cdot)$, and length penalty/reward, $\mathrm{LengthPen}(\cdot)$ yield the estimated joint likelihood P_{Beam}. Finally in *end detection*, OUTPUTSTEP must check for the stop token, <eos>, which may be proposed by $\mathrm{attnCnds}$.

Input-Synchrony Now let us consider the differences when CTC is in the leading role. Note that this algorithm extends Hannun et al. (2014)'s CTC beam search algorithm to include joint scoring with attentional likelihoods. INPUTSTEP performs *hypothesis expansion* by computing CTC's alignment posterior at time step t, $P_{\text{CTC}}(z_t|X)$. Unlike in output-synchrony, here each hypothesis expansion also consumes one step of the input. The same pre-beam size, p, is used to select top candidate alignment units, $\mathrm{CTCCnds}$, but partial hypotheses are only expanded for non-blank and non-repeat candidates. The *joint scoring* block is identical to output-synchrony except for one difference: CTC likelihood, $\mathrm{CTCScore}(\cdot)$, is applied over the full input, $X_{1:T}$, in OUTPUTSTEP and over the partial input, $X_{1:t}$, in INPUTSTEP. This difference engenders a speed vs. accuracy trade-off, which we discuss in D2 of §10.7. Finally, *end detection* simply occurs when all input units have been consumed ($t = T$). Therefore, INPUTSTEP does not require checking for the stop token as all hypotheses at time T are ended.

We propose this particular form of input-synchronous beam search in order to exactly mirror the functions of its output-synchronous counterpart; without this mirrored formulation, we cannot attribute differences in decodings to the swapped roles of CTC and attention. For instance, now we can answer questions such as *can CTC perform hypothesis expansion on par with attention*, allowing us to address concerns about applying weaker joint CTC models during decoding per L2 and L3 in §10.3. To the best of our knowledge, we are the first to examine the theoretical and empirical differences of input and output synchrony through a unified formulation, as discussed further in §10.7. Other forms of input-synchronous beam search in prior works cannot directly be used for this purpose. Triggered Attention (Moritz et al., 2019) is one such example which is purpose-fit for streaming to a degree where several core components (e.g. look-ahead and re-triggering) cannot trivially be re-factored into an output-synchronous variant.

		Model Type			MT			ST		
#	Model Name	Joint Train?	Joint Decode?	Decoding Method	IWSLT14 De-En	IWSLT14 Es-En	MTedX All-En	MuST-C-v2 En-De	MuST-C-v2 En-Ja	MTedX All-En
1	Pure-Attn (Prior)	✗	✗	Attn Only	(32.2)†	(39.0)†	-◇	25.8‡	12.4‡	-◇
2	Pure-Attn (Ours)	✗	✗	Attn Only	32.8 (33.7)	39.0 (39.9)	25.6	27.8	14.3	22.7
3	Joint CTC/Attn	✓	✗	CTC Only	27.3	33.8	22.4	24.4	10.2	21.4
4	Joint CTC/Attn	✓	✗	Attn Only	33.6	39.5	28.0	28.3	14.2	23.7
5	Joint CTC/Attn	✓	✓	Joint I-Sync	33.7	39.7	27.8	**29.2**	15.1	**25.1**
6	Joint CTC/Attn	✓	✓	Joint O-Sync	**34.1**	**39.9**	**28.1**	**29.2**	**15.3**	**25.1**

Table 10.2: Test set performances, as measured by BLEU (↑), of our proposed joint CTC/Attention models compared to pure-attention baselines. Joint CTC/Attention models are always jointly trained, but can be either jointly decoded using input/output synchrony or decoded using only their CTC or attention branches. For IWSLT14, we mention (*tokenized BLEU*) for comparison with prior works: †Raunak et al. (2020) and ‡Inaguma et al. (2020). ◇Prior MTedX works show only All-All or pair-wise settings.

10.5 Experimental Setup

Data We examine the efficacy of our proposed approaches on two language pairs for each of the MT and ST tasks. For MT, we use German-to-English (De-En) and Spanish-to-English (Es-En) from IWSLT14 (Cettolo et al., 2012). For ST, we use English-to-German (En-De) and English-to-Japanese (En-Ja) from MuST-C-v2, reporting tst-COMMON results (Di Gangi et al., 2019). We also examine the multilingual setting of 6 European languages to English (All-En) from MTedX (Salesky et al., 2021) for both tasks. Full dataset descriptions for reproducibility are in §10.11.

Modeling We compare our joint CTC/Attention models to purely attentional encoder-decoder baselines. All proposed and baseline models were tuned separately, using validation sets only, within the same hyperparameter search spaces for training and decoding to ensure fair comparison. All experiments were conducted using ESPnet-ST (Inaguma et al., 2020). Full descriptions of model sizes, hyperparameters, and pre-processing are in §10.11.[3]

Evaluation: Unless otherwise indicated, we measure performance with detokenized case-sensitive BLEU (Post, 2018) on punctuated 1-references.[4]

[3]We compare our baselines for MuST-C-v2 to the default recipes in ESPnet in Table 10.2. For back-compatibility with additional prior works using MuST-C-v1 En-De, see §10.10.

[4]Evaluation with additional metrics is provided in §10.10.

10.6 Results and Analyses

In this section, we first present our main results on 6 benchmark MT and ST tasks. We then present evidence that hierarchical encoding (§10.4) produces stable encoder representations that simplify the decoder's source attention (addressing L1 in §10.3). Next we present evidence that joint decoding is beneficial despite the fact that CTC-only performance lags behind that of attention-only (addressing L2 in §10.3). Finally, we present evidence that CTC's alignment information alleviates attention's end-detection problem in both input and output synchronous joint decoding (§10.4) (addressing L3 in §10.3).

Joint CTC/Attention Models Outperform CTC-only and Attention-only Baselines

As shown in Table 10.2, joint CTC/Attention with output-synchronous decoding outperforms pure-attention across *all* MT and ST tasks (line 2 vs. 6). Joint training while only decoding with the attention branch still outperforms pure-attention models without any joint training (line 2 vs. 4). Note that *CTC is consistently the weaker* of the two branches in jointly trained models (line 3 vs. 4). Joint input/output-synchronous decodings yield further improvements overall, confirming that *both joint training and decoding are beneficial* (line 4 vs. 5/6). However, we find that input-synchrony lags behind output-synchrony (line 5 vs. 6); this phenomenon is discussed further in §10.7.

Hierarchical Encoding Reduces Attention's Alignment Burden

We examine the regularization effect that CTC joint training has on the attentional decoder, per L1 in §10.3, by first quantifying the monotonicity, m of a (L, T) shaped source attention pattern, A:

$$m = \Big(\sum_{2<l\leq L} [\operatorname*{argmax}_{t\in T} A_l \geq \operatorname*{argmax}_{t\in T} A_{l-1}] \Big) / L$$

where $[\cdot]$ denotes the Iverson bracket. In other words, we define monotonicity m as the rate at which the decoder at step l attends most sharply on an input index, $\operatorname{argmax}_{t\in T} A_l$, which is greater than or equal to that of the previous step $l-1$, $\operatorname{argmax}_{t\in T} A_{l-1}$. We compute m over all examples in our validation sets for De-En MT and En-De ST and show the layer-wise averages over all examples and attention heads in Figure 10.2. It can be seen that *the decoder source attention patterns are more monotonic* when using jointly trained hierarchical encoders. Per line 2 of Table 10.1, we argue that this greater monotonicity allows the decoder to more rapidly learn soft alignment patterns – ultimately this advantage is reflected in the overall performance gains observed from joint

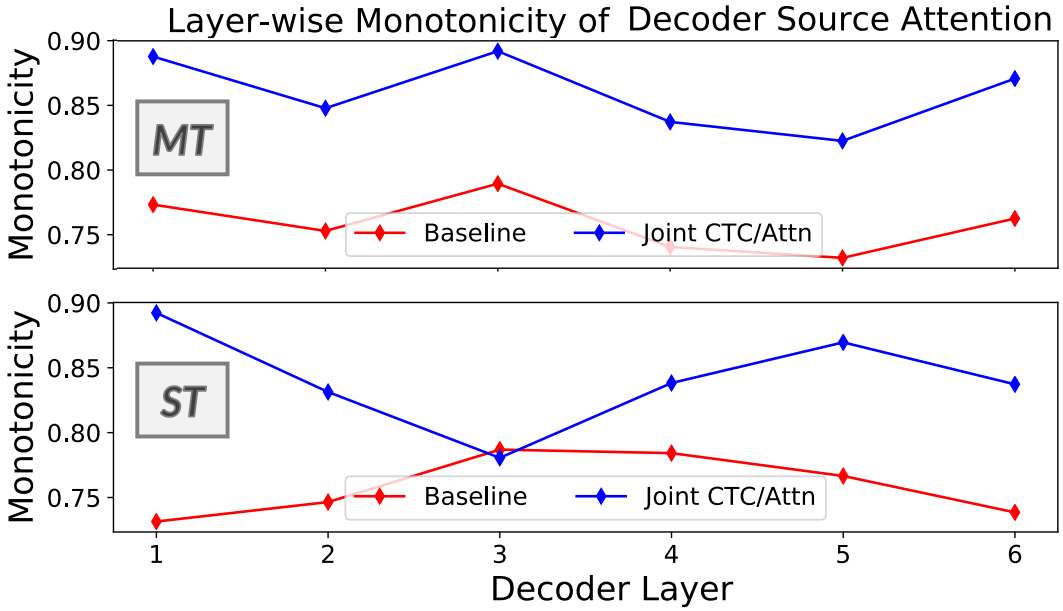

Figure 10.2: Layer-wise monotonicity (↑) of the source-attention patterns produced by MT/ST decoders.

training without joint decoding (line 2 vs. 4 in Table 10.2).

For a qualitative example illustrating the increased monotonicity of decoder source attention patterns, please see §10.10. We also found that *increased monotonicity leads improves multilingual parameter sharing* in our All-En MT and ST models, suggesting that the target-orientation of our encoder reduced the decoder's burden of soft-aligning target English outputs to source languages with varying word-orders (discussed further in §10.10).

What are the respective contributions of SrcCTC and TgtCTC? TGTCTC holds elevated importance as joint decoding is not possible without it. However, we'd like to understand how each component contributes to the benefits observed from joint training without joint decoding in §10.6. In Table 10.3, we ablate SRCCTC and TGTCTC in order to confirm that both contribute to performance gains. Note that SRCCTC on its own appears to contribute more to MT than it does to ST, suggesting that the length adjustment stage is more critical in MT.

Even Weak CTC Models Strengthen Joint CTC/Attention Models

We examine the generalization effect that augmenting autoregressive likelihoods with conditionally independent likelihoods has during inference, per L2 in §10.3, by evaluating De-En MT and En-De ST models on out-of-domain EuroParl test sets (Iranzo-Sánchez et al., 2020). As shown in Table 10.4, joint CTC/Attention models outperform pure-attention baselines across in-domain

SrcCTC	TgtCTC	MT (De-En) IWSLT14	ST (En-De) MuST-C-v2
✗	✗	32.1	27.7
✓	✗	34.1	27.8
✗	✓	33.3	28.1
✓	✓	**34.8**	**28.3**

Table 10.3: Ablation on the impacts of SrcCTC and TgtCTC CTC components of hierarachical encoding, as measured by performance on validation sets. Only attention is used in decoding to enable fair comparisons.

Model	Decoding Method	MT (De-En) In-D	MT (De-En) Out-D	ST (En-De) In-D	ST (En-De) Out-D
Pure-Attn	Attn Only	32.8	15.8	27.8	20.5
Joint C/A	Attn Only	33.6	17.1	28.3	21.0
	+CTC Rescore	33.6	17.1	28.3	21.0
Joint C/A	Joint O-Sync	**34.1**	**17.6**	**29.2**	**21.7**
Joint C/A	CTC Only	27.3	13.1	24.4	16.5
	+Attn Rescore	29.5	13.9	26.2	17.8
Joint C/A	Joint I-Sync	**33.7**	**17.4**	**29.2**	**21.1**

Table 10.4: In/out domain test performances of joint CTC/attention models with various decoding methods.

(In-D) and out-of-domain (Out-D) settings. When decoding only the CTC branch of joint models (denoted as CTC I-Sync in the table) performance is significantly degraded compared to the attention branch of the same models (denoted as Attn O-Sync in the table). This gap appears slightly lessened in the out-of-domain setting where CTC's conditional independence may offer some robustness. Nonetheless these weak CTC models still boost their stronger attention counterparts during joint decoding (both via input and output-synchrony), suggesting that *ensembling of conditionally independent and dependent likelihoods is a powerful technique*.

Further, synchronous joint decoding methods outperform their two-pass re-scoring counterparts (discussed in D2 of §10.7), suggesting that *joint selection of the hypothesis set is necessary* for easing the respective weaknesses of autoregressive and conditionally independent likelihood estimation.

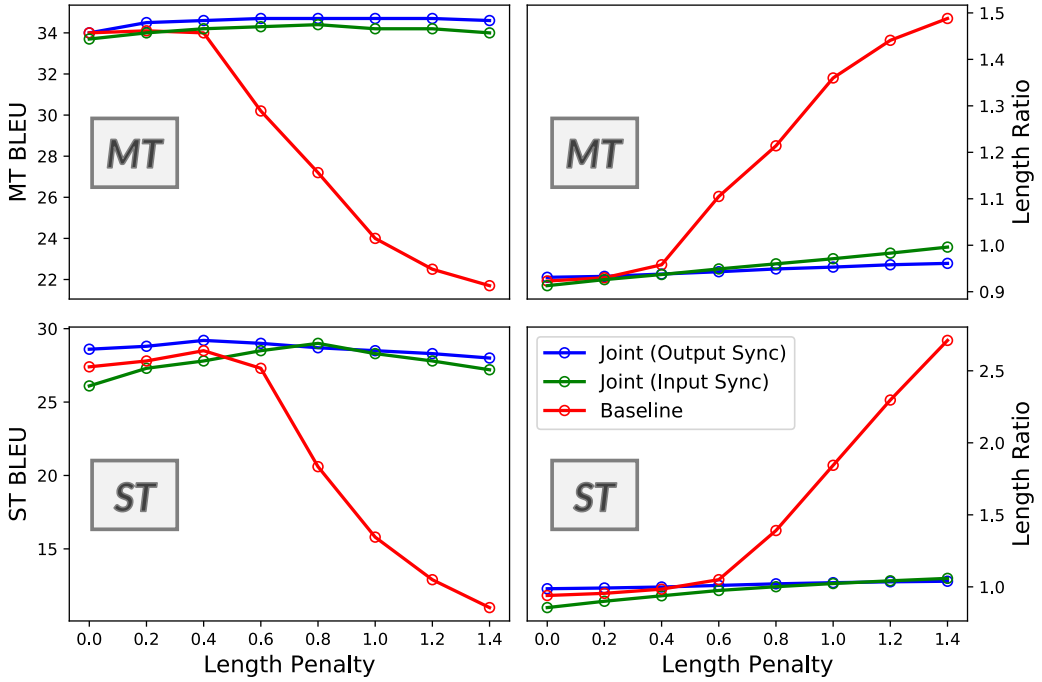

Figure 10.3: Elasticity of BLEU and length ratios ($|\text{hyp}|/|\text{ref}|$) w.r.t length penalty in validation sets.

CTC's Alignment Information Resolves Attention's End-Detection Problem

Finally, we examine the effect that CTC's alignment information has on end detection during decoding, per L3 in §10.3. In Figure 10.3, we observe the change in translation quality (as measured by BLEU) and output length (as measured by hypothesis-to-reference length ratio) when the length penalty (denoted as $\text{LengthPen}(\cdot)$ in Algorithms 1 and 2) is gradually increased, forcing decodings to produce longer outputs. Pure-attention baselines rapidly degenerate when forced to produce hypotheses that are longer than references as they struggle to detect the ends of hypotheses (Murray and Chiang, 2018b). On the other hand, joint decoding produces gradually longer outputs regardless of whether CTC is in a primary role (input-synchrony) or a secondary role (output-synchrony), demonstrating that *CTC alignments ease the decoder's end-detection problem* by explicitly or implicitly ruling out hypotheses of incorrect lengths.

10.7 Discussion: More on Joint Decoding

D1 *Why do input vs. output-synchronous joint decodings yield slightly different results?*
By comparing the CTC likelihood estimation in InputStep vs. OutputStep, it can be

DECODING TYPE		ACCURACY		SPEED
Method	Beam Size	BLEU	Search Error	RTF
Joint O-Sync	5	29.1	0.73%	0.9
Joint O-Sync	10	29.2	0.44%	1.7
Joint O-Sync	50	29.0	0.36%	9.0
Joint I-Sync	5	28.1	1.02%	0.4
Joint I-Sync	10	28.6	1.09%	0.9
Joint I-Sync	50	29.0	0.87%	6.4

Table 10.5: Speed vs. accuracy for joint input/output-sync decoding of En-De ST val. set as a fxn. of beam size.

seen that there is a trade-off between speed vs. accuracy. First, note that in OUTPUTSTEP, $\mathrm{CTCScore}(y_{1:l}, X_{1:T})$, is a marginalization over the likelihoods of all possible alignments of the partial hypothesis $y_{1:l}$ to the full input $X_{1:T}$ (Seki et al., 2019a). On the other hand, $\mathrm{CTCScore}(\tilde{y}, X_{1:t})$ in INPUTSTEP is an estimation of the marginalized likelihoods of the partial hypothesis $y_{1:l}$ to the *partial* input $X_{1:t}$ (Graves and Graves, 2012; Hannun et al., 2014). Even at step T, these two $\mathrm{CTCScore}(\cdot)$'s are not equivalent. Since $\mathrm{CTCCnds}$ may include the blank token, INPUTSTEP may prune partial hypotheses at a previous beam step which would have merged with $y_{1:l}$. Therefore, $\mathrm{CTCScore}(\cdot)$ in *input-synchrony is less accurate*. However, *input-synchrony requires fewer computations*. Using dynamic programming, output-synchrony computes $\mathrm{CTCScore}(\cdot)$ for all partial hypothesis within a single beam step with $\mathcal{O}(bpT)$ log-additions (Watanabe et al., 2017b) while input-synchrony uses only $\mathcal{O}(bp)$ log-additions (Hannun et al., 2014).

In Table 10.5, we perform an experimental validation of our theoretical understanding of the speed vs. accuracy trade-off between the two synchronous joint decoding variants. To quantify speed, we compute the real-time factor (RTF) as the ratio of decoding time over the duration of input speech. To quantify accuracy beyond the BLEU metric, we compute the search error rate (Meister et al., 2020) by counting the sequences for which the hypothesis has higher exact likelihood than the reference. *For the same beam size, output is slower but more accurate than input-synchronous.* We conclude that input-synchrony may in fact be preferable in applications with latency constraints.

D2 *Why did synchronous joint decodings outperform re-scoring decodings in Table 10.4?*

There is a family of two-pass decoding algorithms (Watanabe et al., 2017b; Sainath et al., 2019), which also achieve joint decoding by first estimating the likelihoods of a subset of sequences

$\mathcal{V}'$ with one module and then re-scoring the estimates with the other module. In these approaches, the subset $\mathcal{V}'$ is determined asynchronously, meaning the joint likelihood is not considered until the re-scoring step; this delayed consideration of the joint likelihood is the main drawback compared to the synchronous approaches. If the attentional decoder is used to determine $\mathcal{V}'$, then $\mathcal{V}'$ would suffer from exposure/label bias and the length problem (§10.2). On the other hand, if CTC is used to determine $\mathcal{V}'$, the lack of causal modeling in CTC leads to poor estimates of $\mathcal{V}'$ – particularly for translation.

10.8 Related Works

The idea of using latent alignments to improve autoregressive translation has been explored previously by Haviv et al. (2021) who concluded that CTC alignments are not compatible with teacher forcing. The key difference is that we train CTC and autoregressive models jointly while Haviv et al. (2021) sought to apply CTC to train autoregressive models, replacing cross-entropy entirely. More recently in a concurrent work, Zhang et al. (2022a) have also shown the effectiveness of jointly training CTC and attention in the context of ST for un-written languages where no ASR transcriptions are available. We believe that our contribution showing the effectiveness of also jointly decoding CTC and attention demonstrates an additional technique which can further improve their direction. Our work also differs in that we seek to incorporate the ASR objective into ST via hierarchical encoding.

Other concurrent works integrated CTC and attention within blockwise streaming (Deng et al., 2022a) and compositional multi-decoder (Yan et al., 2022a) architectures for ST in particular. Our work supports their findings by addressing *why* CTC is helping, and we provide a unified approach that generalizes to both MT and ST. Prior works have also used the non-autoregressive property of CTC as means for speeding up autoregressive models during inference (Inaguma et al., 2021a; Gaido et al., 2021), but these works focus on latency and do not apply CTC to improve translation quality.

10.9 Conclusion

We propose to jointly train and decode CTC/attention models for MT and ST using 1) hierarchical encoding to resolve incompatibilities between CTC and the non-monotonic mappings in translation and 2) synchronous decoding to ease the exposure/label biases of autoregressive decoders with CTC's conditionally independent alignment information. Our analyses reveal several reasons why even weak CTC models benefit autoregressive translation via joint modeling, suggesting that future explorations into jointly modeling attentional decoders with other latent alignment models (Graves

Decoding Type		Speed	
Method	Beam Size	RTF	%Δ
Pure-Attn O-Sync	5	0.9	-
Pure-Attn O-Sync	10	1.2	-
Pure-Attn O-Sync	50	3.5	-
Joint CTC/Attn O-Sync	5	0.9	+0%
Joint CTC/Attn O-Sync	10	1.7	+42%
Joint CTC/Attn O-Sync	50	9.0	+157%
Joint CTC/Attn I-Sync	5	0.4	-56%
Joint CTC/Attn I-Sync	10	0.9	-25%
Joint CTC/Attn I-Sync	50	6.4	+85%

Table 10.6: **Limitations Table:** comparison of joint decoding and pure-attention RTFs across different beam sizes. %Δ between the joint RTF and pure-attention RTF for the same beam size is shown, where positive %'s indicate slow-downs and negative %'s indicate speed-ups.

and Graves, 2012; Ghazvininejad et al., 2020; Saharia et al., 2020) may uncover similar benefits.

We've now demonstrated a sequential, temporal multi-sequence approach. Next we'll evaluate the efficacy in a streaming setting.

Limitations

There are several potential limitations pertaining to the increased computational overhead and latency of the joint modeling approach. One concern is that joint decoding is much slower, but we found that input-synchronous joint decoding is actually faster than pure-attention decoding for smaller beam sizes, as shown in Table 10.6.

The other limitation is that our MT models upsample input representations in the early layers of the encoder, thereby increasing the computations in subsequent encoder layers and the decoder's cross-attention. We can use LegoNN-based encoders (Dalmia et al., 2022) to adjust the up-sampling rate to a fractional value, minimizing the computations given dataset statistics. Alternatively, we may avoid the need for up-sampling by applying a larger byte-pair encoding size (Kudo and Richardson, 2018a) to the target language compared to the source language. CTC's use in guiding efficient down-sampling of representations in ST (Gaido et al., 2021) suggest that it may also be applied for efficient up-sampling for MT – we leave this study on efficiency to future work.

Model Name	Decoding Method	IWSLT14 (De-En) BLEU (↑)	IWSLT14 (De-En) TER (↓)	IWSLT14 (De-En) METEOR (↑)	MuST-C-v2 En-De BLEU (↑)	MuST-C-v2 En-De TER (↓)	MuST-C-v2 En-De METEOR (↑)
Pure-Attn (Ours)	Attn-only	32.8	50.9	29.4	27.8	59.1	38.6
Joint CTC/Attn	Attn-only	33.6	50.7	30.0	28.3	58.4	39.2
Joint CTC/Attn	Joint I-Sync	33.7	50.6	30.0	**29.2**	57.8	40.1
Joint CTC/Attn	Joint O-Sync	**34.1**	**49.9**	**30.2**	**29.2**	**57.5**	**40.2**

Table 10.7: Test set performances, as measured by BLEU (↑), TER (↓), and METEOR (↑), of our proposed joint CTC/Attention models compared to pure-attention baselines.

Model Name	Decoding Method	IWSLT14 De-En	IWSLT14 Es-En	MuST-C-v2 En-De	MuST-C-v2 En-Ja
Pure-Attn (Ours)	Attn O-sync	34.1	41.2	28.5	11.3
Joint CTC/Attn	Joint I-sync	34.6	42.0	29.0	12.4
Joint CTC/Attn	Joint O-sync	35.0	42.3	29.2	12.4

Table 10.8: Valid set performances, as measured by BLEU (↑).

Finally, note that the corpora used for the MT experiments in this work are considered medium resourced. Prior work (Murray and Chiang, 2018b) has shown that the autoregressive end-detection problem exists across low to high resourced scenarios; suggesting that the CTC/attention approach would be generally beneficial. We leave the study of joint CTC/attention modeling on higher resourced MT corpora to future work.

10.10 Supplementary Information

Additional Translation Metrics

To supplement our BLEU evaluation in Table 10.2, we also measure the translation quality of our models using Translation Error Rate (TER) (Snover et al., 2006) and METEOR (Banerjee and Lavie, 2005). As shown in Table 10.7, our findings are consistent across all three metrics for both MT and ST models.

MODEL NAME	MUST-C-V1 En-De
ESPnet-ST[1]	22.9
Dual-Decoder[2]	23.6
E2E-ST-TDA[3]	25.4
Multi-Decoder[4]	26.4
Pure-Attn (ours)	27.1
Joint CTC/Attn w/ Joint O-Sync	**28.2**

Table 10.9: Comparison of our best MuST-C-v1 En-De Joint CTC/Attn model and our Pure-Attn baseline with prior works: [1]Inaguma et al. (2020), [2]Le et al. (2020b), [3]Du et al. (2022), [4]Dalmia et al. (2021b)

MuST-C-v1 Back-Compatibility

See Table 10.9 for results compared to prior works.

Valid Set performances

Table 10.8 presents the validation performances for our ST and MT models.

Description of Encoder Architectures

LegoNN Encoder (Dalmia et al., 2022) is a stacked multi-block architecture that was introduced to encode and re-sample the input sequence into a sequence of representations of a desired length, which is typically a factor of the input sequence. It first encodes the input using transformer encoder blocks (Vaswani et al., 2017) and then re-encodes them into a sequence of latent representations using cross-attention. Starting from a sequence of learnable positional embeddings (Gehring et al., 2017), these latent representations are learned using another stack of transformer encoder layers with an added cross-attention component over the input representations in each block.

The Conformer encoder (Gulati et al., 2020) is a stacked multi-block architecture and has shown consistent improvement over a wide range of E2E speech processing applications (Guo et al., 2021). It includes a multi-head self-attention module, a convolution module, and a pair of position-wise feed-forward modules in the Macaron-Net style. While the self-attention module learns the long-range global context, the convolution module aims to model the local feature patterns synchronously.

Increased Cross-Attentional Monotonicity Leads to Increased Multilingual Parameter Sharing

We further examine the source attention parameters in our All-En models to understand the impact that the increased monotonicity of decoder attention (§10.6) has on multilingual parameter sharing. To do so, we extract sparse subnets for each language pair following the Lottery Ticket Sparse Fine-Tuning proposed by Ansell et al. (2022) and compute the pair-wise sharing across the 6 source languages, as measured by the count of overlapping parameters between subnets. In Figure 10.4, we show the relative change ($\Delta\%$) in multilingual sharing when using hierarchical encoding compared to the baseline. The broad increases suggest that the target-orientation of our encoder *reduced the decoder's burden of soft-aligning* target English outputs to source languages with varying word-orders, allowing for more efficient allocation of capacity.

compare_mt.py Length Analysis

As shown in Figure 10.5, both joint synchronous decodings are more robust than pure-attention for long output lengths across both MT and ST. Input-synchrony appears particularly more robust in generation of very long outputs for ST.

View of Regularized Attention

See Figure 10.6 for a qualitative example of monotonic source attention patterns (supplementary to the quantitative monotonicity in Figure 10.2).

10.11 Reproducibility

Dataset Descriptions

See Table 10.10 for dataset descriptions. Data preparation was done using ESPnet recipes.

Model Architectures

See Table 10.11 for model architectures.

Training/Decoding Hyperparameters

See Tables 12-15 for hyperparameter descriptions.

Dataset	Task	Source Lang(s)	Target Lang(s)	Domain	# Train/Valid/Test Utts	# Speech Train Hours
IWSLT17 (Cettolo et al., 2012)	MT	De	En	TED Talk	160k/7k/7k	-
IWSLT17 (Cettolo et al., 2012)	MT	De	Es	TED Talk	160k/7k/7k	-
MuST-C-v2 (Di Gangi et al., 2019)	ASR/ST	En	De	TED Talk	250k/1k/3k	450h
MuST-C-v2 (Di Gangi et al., 2019)	ASR/ST	En	Ja	TED Talk	330k/1k/3k	540h
MTedX (Salesky et al., 2021)	MT	Es, Fr, Pt, It, Ru, El	En	TED Talk	130k/6k/6k	-
MTedX (Salesky et al., 2021)	ASR	Es, Fr, Pt, It, Ru, El	En	TED Talk	400k/6k/6k	730h
MTedX (Salesky et al., 2021)	ST	Es, Fr, Pt, It, Ru, El	En	TED Talk	130k/6k/6k	250h
EuroParl (Iranzo-Sánchez et al., 2020)	MT	De	En	Parliament Speech	- /-/2k	-
EuroParl (Iranzo-Sánchez et al., 2020)	ST	En	De	Parliament Speech	-/-/1k	-

Table 10.10: MT/ST/ASR dataset descriptions. Utterance counts are rounded to the nearest thousand. Language codes: De=German, En=English, Es=Spanish, Ja=Japanese, Fr=French, Pt=Portuguese, It=Italian, Ru=Russian, El=Greek

Model	Task	# Encoder Layers [$\mathcal{S}$]	# Decoder Layers	SrcCTC Layer	Up/Down-Sample	Pre-Train Init	Src BPE Size	Tgt BPE Size	# Params
Pure-Attn	MT	12 [6,12,18]	6	-	-	-	10k (joint)		54M
Joint CTC/Attn	MT	18 [6,12,18]	6	6	3x	-	10k (joint)		95M
Pure-Attn	ST	18 [12, 18]	6	-	1/4x	Enc lyr 1-12 from ASR	4k	4k	74M
Joint CTC/Attn	ST	18 [12, 18]	6	12	1/4x	Enc lyr 1-12 from ASR	4k	4k	72M
Pure-Attn	ASR	12	6	-	-	-	4k	4k	46M

Table 10.11: MT/ST/ASR model descriptions. The best MT/ST Encoder layers settings were selected over a search space indicated by $\mathcal{S}$. Parameter counts are rounded to the nearest million. Note that the 12 layer pure-attn model outperformed the 18 layer version and that the 12 layer joint model still outperformed these baselines.

Metrics

Sacrebleu signature for all non-Japanese:

`BLEU+case.mixed+numrefs.1 +smooth.exp+tok.13a+version.1.5.1`

Sacrebleu signature for Japanese:

`BLEU+case.mixed+lang.en-ja+numrefs.1 +smooth.exp+tok.ja-mecab-0.996-IPA +version.1.5.1`

For tokenized BLEU in the IWSLT MT datasets we used `mutibleu.perl` (Moses-SMT, 2018)

Computing

ST models were trained on 2 x V100 for 2 days. MT models were trained on 1 x A6000 for 1 day.

Hyperparameter	Value
Hidden Dropout	0.3
Attention dropout	0.3
Activation dropout	0.3
LR schedule	inv. sqrt. (Vaswani et al., 2017)
Max learning rate	best of [1e-3, 3e-3]
Warmup steps	10000
Number of steps	200 epoch
Adam eps	1e-9
Adam betas	(0.9, 0.98)
Weight decay	1e-4
$\lambda 1$, $\lambda 2$	(1, 2)

Table 10.12: Training Hyperparameters for MT Models.

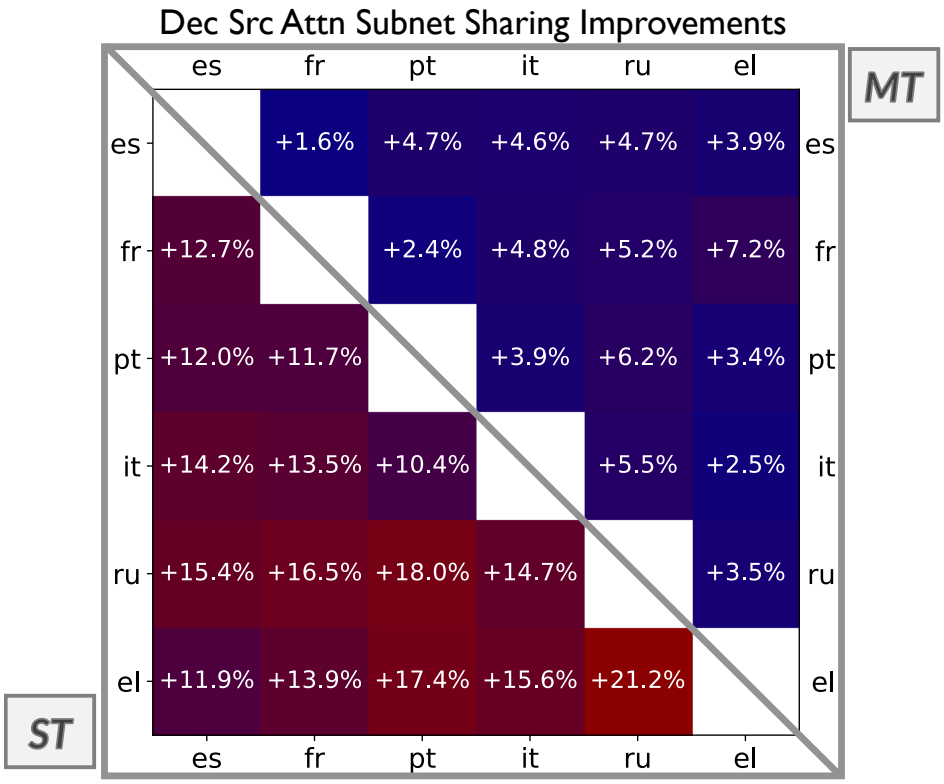

Figure 10.4: Improvement of multilingual sharing in MT/ST decoder source attention parameters when using joint CTC/Attention vs. attention-only training, as measured by pair-wise $\Delta\%$ in sparse subnet overlap.

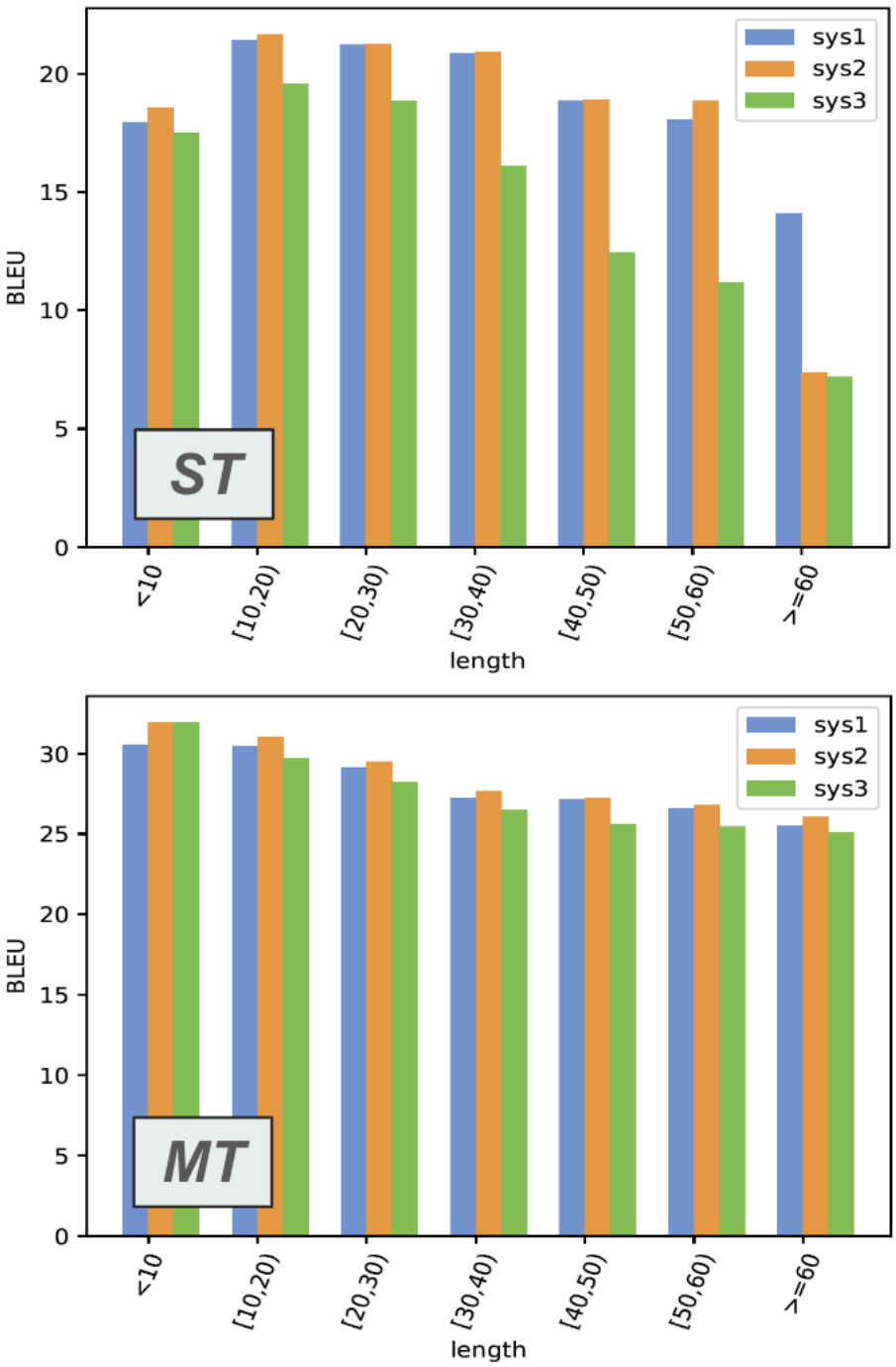

Figure 10.5: Compare-mt (Neubig et al., 2019) output sentence length to BLEU for joint decoding vs pure-attention models. Model codes: sys1 = Joint Input-Sync, sys2 = Joint Output-Sync, sys3 = Pure-Attn

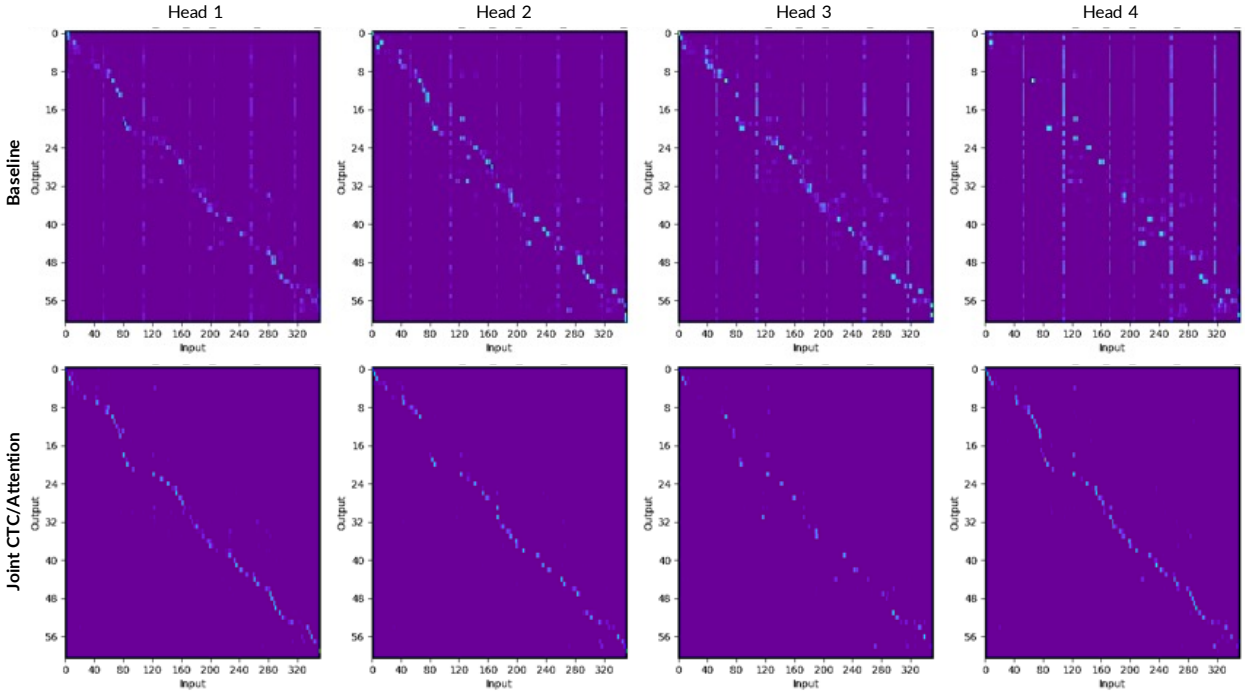

Figure 10.6: Visualization of source attention patterns produced by pure-attention baseline (top) vs. joint CTC/attention (bottom) ST models. Qualitative example extracted from the final decoder layer. Irregular patterns are observable in the pure-attention plots, but not in the joint CTC/attention plots.

Hyperparameter	Value
Hidden Dropout	0.1
Attention dropout	0.1
Activation dropout	0.1
LR schedule	inv. sqrt. (Vaswani et al., 2017)
Max learning rate	0.002
Warmup steps	25000
Number of steps	40 epoch
Adam eps	1e-9
Adam betas	(0.9, 0.98)
Weight decay	0.0001
$\lambda 1$, $\lambda 2$	(2, 5)

Table 10.13: Training Hyperparameters for ST Models.

Decoding Type	Hyperparameter	Value
Pure Attn	Max Length Ratio	[1, 1.2, 1.4, 1.6, 1.8, 2, 2.5, 3]
	Penalty	[0, 0.2, 0.4, 0.6, 0.8, 1.0]
	Beam Size	5
Joint O-Sync	Max Length Ratio	1
	Penalty	[0, 0.2, 0.4, 0.6, 0.8, 1.0]
	CTC Weight	0.3
	Beam Size	5
Joint I-Sync	Max Length Ratio	1
	Penalty	[0, 0.2, 0.4, 0.6, 0.8, 1.0]
	Blank Penalty	[0.5, 0.75, 1.0]
	CTC Weight	[0.3, 0.5]
	Beam Size	[10, 30]

Table 10.14: Decoding Search Space MT Models.

Decoding Type	Hyperparameter	Value
Pure Attn	Max Length Ratio	1
	Penalty	[0,0.2,0.4,0.6,0.8,1.0]
	Beam Size	[10, 30, 50]
Joint O-Sync	Max Length Ratio	1
	Penalty	[0,0.2,0.4,0.6,0.8,1.0]
	CTC Weight	[0.3, 0.5]
	Beam Size	[10, 30, 50]
Joint I-Sync	Max Length Ratio	1
	Penalty	[0,0.2,0.4,0.6,0.8,1.0]
	Blank Penalty	1
	CTC Weight	[0.3, 0.5]
	Beam Size	[10, 30, 50]

Table 10.15: Decoding Search Space ST Models.

Chapter 11

Application of *Hierarchical CTC* to Streaming Speech Translation

Summary

In this chapter, we apply the Hierarchical CTC approach described in Chapter 8, a **sequential, temporal** multi-sequence model, to a streaming setting.

This chapter describes CMU's submission to the IWSLT 2023 simultaneous speech translation shared task for translating English speech to both German text and speech in a streaming fashion. We first build offline speech-to-text (ST) models using the joint CTC/attention framework. These models also use WavLM front-end features and mBART decoder initialization. We adapt our offline ST models for simultaneous speech-to-text translation (SST) by 1) incrementally encoding chunks of input speech, re-computing encoder states for each new chunk and 2) incrementally decoding output text, pruning beam search hypotheses to 1-best after processing each chunk. We then build text-to-speech (TTS) models using the VITS framework and achieve simultaneous speech-to-speech translation (SS2ST) by cascading our SST and TTS models.

11.1 Introduction

In this chapter, we present CMU's English to German simultaneous speech translation systems. Our IWSLT 2023 (Agarwal et al., 2023) shared task submission consists of both simultaneous speech-to-text (SST) and simultaneous speech-to-speech (SS2ST) systems. Our general strategy is to first build large-scale offline speech translation (ST) models which leverage unpaired speech data, ASR data, and ST data. We then adapt these offline models for simultaneous inference. Finally, we use a text-to-speech model to achieve SS2ST in a cascaded manner.

In particular, our system consists of:

1. Offline ST using joint CTC/attention with self-supervised speech/text representations (§11.3)
2. Offline-to-online adaptation via chunk-based encoding and incremental beam search (§11.3)
3. Simultaneous S2ST by feeding incremental text outputs to a text-to-speech model (§11.3)

11.2 Task Description

The IWSLT 2023 simultaneous speech translation track[1] is a shared task for streaming speech-to-text and speech-to-speech translation of TED talks. This track mandates that systems do not perform re-translation, meaning that the streaming outputs cannot be edited after the system receives more input audio. Systems are required to meet a particular latency regime: SST systems must have <2 seconds average lagging (AL) and SS2ST systems must have <2.5 seconds start offset (SO) (Ma et al., 2020b).

Of the allowed training data, we selected a subset of in-domain data to train our ASR and ST models: for ASR we use TEDLIUM v1 and v2 (Zhou et al., 2020a) and for ST we used MuST-C v2 (Di Gangi et al., 2019). We also use a set of cross-domain data to train our MT and TTS models due to the lack of in-domain data: for MT we use Europarl, NewsCommentary, OpenSubtitles, TED2020, Tatoeba, and ELRC-CORDIS News (Tiedemann et al., 2020). For TTS we use CommonVoice (Ardila et al., 2020). The following section describes how each of the ASR, ST, MT, and TTS components fit together in our ultimate systems.

11.3 System Description

Offline Speech Translation (ST)

As shown in Figure 11.1, our offline ST models are based on the joint CTC/attention framework (Watanabe et al., 2017b; Yan et al., 2023a). Compared to a purely attention-based approach, joint CTC/attention has been shown to reduce the soft-alignment burden, provide a positive ensembling effect, and improve the robustness of end-detection during inference (Yan et al., 2023a).

To leverage unpaired speech data, we use first use WavLM representations (Chen et al., 2022) as front-end features to train ASR models. In these models, a pre-encoder module (Chang et al., 2021) applies feature dimension down-sampling and a learned weighted combination of WavLM layers before feeding to a Conformer encoder (Gulati et al., 2020). The pre-encoder and encoder modules from ASR are then used to initialize our ST models.

[1] https://iwslt.org/2023/simultaneous

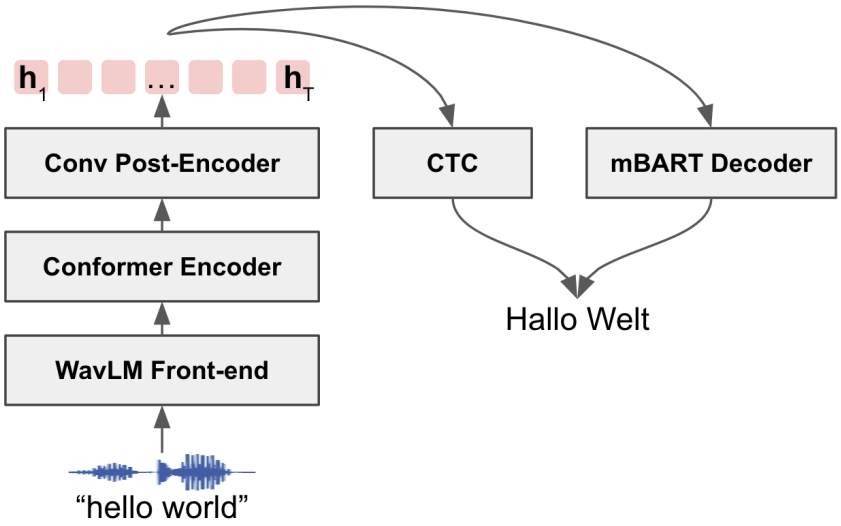

Figure 11.1: Offline ST model architecture based on the joint CTC/attention framework with a WavLM front-end and mBART decoder.

To leverage unpaired text data, we use the mBART decoder (Tang et al., 2020) as an initialization for our ST models. Following (Li et al., 2020), we freeze all feed-forward layers during fine-tuning and use a post-encoder down-sampling layer to reduce the computational load.

We fine-tune our ST models using the following interpolated loss function: $\mathcal{L} = \lambda_1 \mathcal{L}_{\text{ASR\_CE}} + \lambda_2 \mathcal{L}_{\text{ASR\_CTC}} + \lambda_3 \mathcal{L}_{\text{ST\_CE}} + \lambda_4 \mathcal{L}_{\text{ST\_CTC}}$. Here, the cross-entropy (CE) losses are used to train attentional decoders. Note that in Figure 11.1, we omit the ASR attentional decoder and CTC components as these function as training regularizations and do not factor into the inference proceedure. We perform fine-tuning on in-domain data consisting primarily of MuST-C (Di Gangi et al., 2019).

To leverage additional in-domain data, we apply MT pseudolabeling to TEDLIUM ASR data (Zhou et al., 2020a). We also use the same MT model to apple sequence-level knowledge distillation to the MuST-C data. The MT model is a pre-trained DeltaLM-large (Ma et al., 2021) fine-tuned on the corpora listed in Section 11.2. The pseudo-labels and distilled sequences were then translated from English to German using a beam size of 10.

Simultaneous Speech Translation (SST)

We adapt our offline ST model for streaming inference by using a chunk-based processing of input speech. As shown in Figure 11.2, our scheme uses a fixed duration (e.g. 2 seconds) to compute front-end and encoder representations on chunks of input speech. With each new chunk, we recompute front-end and encoder representations using the incrementally longer input speech.

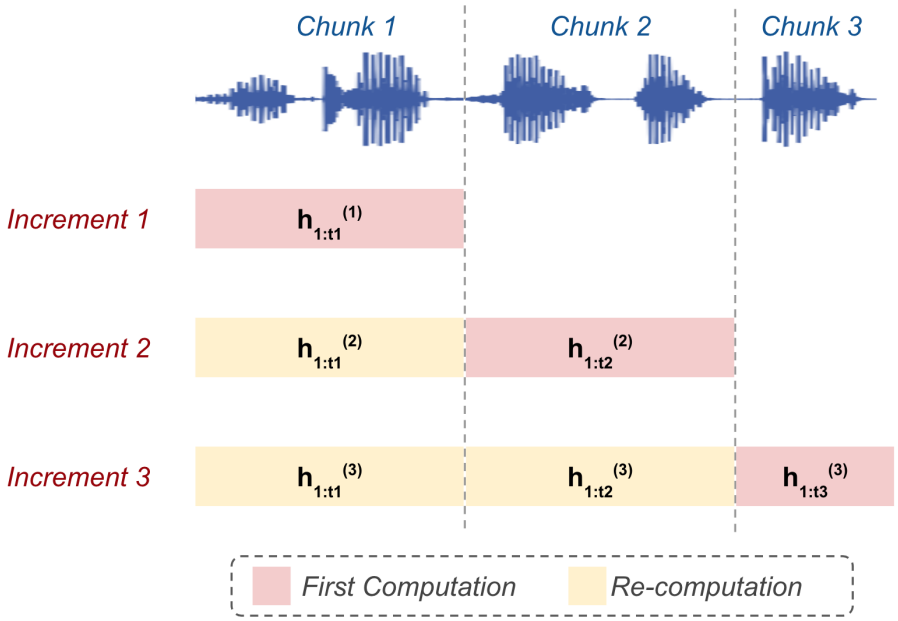

Figure 11.2: Incremental encoding strategy which processes chunks of input speech by re-computing representations corresponding to earlier chunks.

To produce incremental translation outputs, we apply several modifications to the offline joint CTC/attention beam search. As shown in Algorithm 1, we run beam search for each chunk of input. Unless we know that the current chunk is the final chunk, we perform end-detection using the heuristics introduced by (Tsunoo et al., 2021). If any of the hypotheses in our beam propose a next candidate which is the special end-of-sequence token or a token which already appeared in the hypothesis, then this strategy determines that the outputs have likely covered all of the available input. At this point, the current hypotheses should be considered unreliable and thus the algorithm rewinds hypotheses to the previous step.

After the end has been detected within the current chunk, we prune the beam to the 1-best hypothesis and select this as our incremental output – this pruning step is necessary to avoid re-translation. When the next input chunk is received, beam search continues from this 1-best hypothesis.

Simultaneous Speech-to-Speech Translation (S2ST)

Simultaneous S2ST model is created by feeding incremental text outputs to a German text-to-speech model. We use end-to-end TTS model VITS (Kim et al., 2021) and train a single speaker German TTS model using CommonVoice dataset(Ardila et al., 2020). VITS consists of text-

Algorithm 2 Beam search step with rewinding of unreliable hypotheses on non-final chunks and incremental pruning upon end-detection.

procedure BEAMSTEP(hyps, prevHyps, isFinal)
  newHyps $= \{\}$; endDetected $=$ False
  for $y_{1:l-1} \in$ prtHs **do**
    attnCnds $=$ top-k$(P_{\text{Attn}}(y_l|X, y_{1:l-1}), \text{k} = p)$
    for $c \in$ attnCnds **do**
      $y_{1:l} = y_{1:l-1} \oplus \text{c}$
      $\alpha_{\text{CTC}} = \text{CTCScore}(y_{1:l}, X_{1:T})$
      $\alpha_{\text{Attn}} = \text{AttnScore}(y_{1:l}, X_{1:T})$
      $\beta = \text{LengthPen}(y_{1:l})$
      $P_{\text{Beam}}(y_{1:l}|X) = \alpha_{\text{CTC}} + \alpha_{\text{Attn}} + \beta$
      newHyps$[y_{1:l}] = P_{\text{Beam}}(\cdot)$
      if (!isFinal) and (c is <eos> or repeat) **then**
        endDetected $=$ True
        newHyps $=$ prevHyps ▷ rewind
      else if l is max$L$ **then**
        endDetected $=$ True
      end if
    end for
  end for
  if endDetected **then** ▷ incremental pruning
    newHyps $=$ top-k$(P_{\text{Beam}}(\cdot), \text{k} = 1)$
  else ▷ standard pruning
    newHyps $=$ top-k$(P_{\text{Beam}}(\cdot), \text{k} = b)$
  end if
  return newHyps, endDetected
end procedure

encoder, flow based stochastic duration predictor from text, variational auto-encoder for learning latent feature from audio and generator-discriminator based decoder for generating speech from latent feature. We use character as input to the TTS model.

We select a suitable speaker from CommonVoice German dataset and train single speaker TTS. As CommonVoice may contain many noisy utterances which can hurt performance of TTS, we use data-selection for high-quality subset. The data selection process involves identifying the speaker who has the highest number of utterances with high speech quality. To determine the speech quality, we use speech enhancement metric DNSMOS (Reddy et al., 2021) which provides an estimation of the speech quality. We evaluate the speech quality for the top five speakers with the largest number of utterances. To establish the high-quality subset, we set a threshold of 4.0 for selecting sentences that meet the desired quality level. Based on this criterion, we choose the second speaker, who has approximately 12 hours of high-quality data.

Finally, we combine our trained German TTS model with SST module during inference. We feed incremental translation text outputs to TTS and synthesize translated speech.

MODEL	QUALITY	LATENCY	
OFFLINE SPEECH TRANSLATION (ST)	BLEU ↑	-	
Multi-Decoder CTC/Attn (Yan et al., 2023b)	30.1	-	-
WavLM-mBART CTC/Attn (Ours)	32.5	-	-
SIMUL SPEECH TRANSLATION (SST)	BLEU ↑	AL ↓	LAAL ↓
Time-Sync Blockwise CTC/Attn (Yan et al., 2023b)	26.6	1.93	1.98
WavLM-mBART CTC/Attn (Ours)	30.4	1.92	1.99
SIMUL SPEECH-TO-SPEECH TRANSLATION (SS2T)	ASR-BLEU ↑	SO ↓	EO ↓
WavLM-mBART CTC/Attn + VITS (Ours)	26.7	2.33	5.67

Table 11.1: Results of our English to German ST/SST/SS2ST models on MuST-C-v2 tst-COMMON.

11.4 Experimental Setup

Our models were developed using the ESPnet-ST-v2 toolkit (Yan et al., 2023b). Our ST/SST model uses WavLM-large as a front-end (Chen et al., 2022). A linear pre-encoder down-samples from 1024 to 80 feature dim. Our encoder is a 12 layer Conformer with 1024 attention dim, 8 attention heads, and 2048 linear dim (Gulati et al., 2020). A convolutional post-encoder then down-samples along the length dimension by a factor of 2. Our decoder follows the mBART architecture and we initialize using the mBART-large-50-many-to-many model (Tang et al., 2020). Our ST CTC branch uses the same 250k vocabulary as the mBART decoder to enable joint decoding. Our TTS model consists of 6 transformer encoder layers for text-encoder, 4 normalizing flow layers for duration predictor, 16 residual dilated convolutional blocks as posterior encoder and multi-period HiFiGan (Kong et al., 2020) style decoder. We train VITS model for 400 epochs with AdamW (Loshchilov and Hutter, 2019) optimizer.

During inference, we use a chunk size of 2 seconds for SST and 2.5 seconds for SS2ST. For both SST and SS2ST we use beam size 5, CTC weight 0.2, and no length penalty/bonus. To account for incremental outputs which end in a prefix of a word rather than a whole word, we delay outputs for scoring by 1 token. There are two exceptions to this token delay: if the last token is a valid German word or a punctuation, then we do not delay.

We evaluate translation quality using BLEU score (Papineni et al., 2002) for ST/SST and ASR-BLEU score for SS2ST. ST/SST references are case-sensitive and punctuated while SS2ST references are case-insensitive and un-punctuated. The ASR model used for ASR-BLEU is Whisper-

small (Radford et al., 2023). We evaluate translation latency for SST using average lagging (AL) (Ma et al., 2020b) and length-adaptive average lagging (LAAL) (Papi et al., 2022). We evaluate translation latency for SS2ST using start (SO) and end-offset (EO) (Ma et al., 2020b).

11.5 Results

Table 11.1 shows the quality and latency of our SST and SS2ST models as measured on En-De tst-COMMON. We also show the ST performance of our model for reference. As a baseline, we compare to the IWSLT-scale ST and SST systems developed in Yan et al. (2023b) – our systems show improved quality, primarily due to the use of WavLM and mBART self-supervised representations.

From ST to SST, we observe a 6% quality degradation. Note that the average duration of tst-COMMON utterances is around 5 seconds, meaning the corresponding latency gain is 60%. From SST to SS2ST, we observe a 12% quality degradation. Note that both the TTS model and the Whisper ASR model powering the ASR-BLEU metric contribute to this gap.

11.6 Conclusion

We describe our English to German simultaneous speech-to-text and speech-to-speech translation systems for the IWSLT 2023 shared task. We start by building large-scale offline speech-to-text systems which leverage self-supervised speech and text representations. We then adapt these offline models for online inference, enabling simultaneous speech-to-text translation. Finally, we feed streaming text outputs to a down-stream TTS model, enabling simultaneous speech-to-speech translation.

We've now established the efficacy of sequential, temporal multi-sequence modeling for a particular type of streaming speech translation - that is, outputs are incrementally displayed to the user. Previously displayed text is never changed, meaning the display is **static**. Next we consider an alternative type of display which is **dynamic**.

Chapter 12

Post-Edit Operations: Dynamic Display Streaming Speech Translation

Summary

This section delves into a dynamic display method for streaming speech translation which leverages **sequential, temporal** multi-sequence modeling.

We propose to simultaneously generate automatic speech recognition (ASR) and ST predictions such that each source language word is explicitly mapped to a target language word. A major challenge arises from the fact that translation is a *non-monotonic* sequence transduction task due to word ordering differences between languages – this clashes with the *monotonic* nature of ASR. Therefore, we propose to generate ST tokens out-of-order while remembering how to re-order them later. We achieve this by predicting a sequence of tuples consisting of a source word, the corresponding target words, and post-editing operations dictating the correct insertion points for the target word. We examine two variants of such *operation sequences* which enable generation of monotonic transcriptions and non-monotonic translations from the same speech input simultaneously. We apply our approach to offline and real-time streaming models, demonstrating that we can provide *explainable* translations without sacrificing quality or latency. In fact, the delayed reordering ability of our approach improves performance during streaming. As an added benefit, our method performs ASR and ST simultaneously, making it faster than using two separate systems to perform these tasks.

(a)	ASR: I really need it ST : Ich brauche das wirklich
(b)	ASR: I really need it ST : Ich brauche das wirklich
(c)	I Ich [1] really wirklich [4] need brauche [2] it das [3]
(d)	I [NO_OPS] Ich [EOP] really [SET_MARKER] wirklich [EOP] need [JMP_BWD] brauche [EOP] it [NO_OPS] das [EOP]

Figure 12.1: Target sequences for the direct E2E ST models with ASR multi-tasking (a) and our proposed E2E model (absolute positional operation sequence (c) and relative shift operation sequence (d)) generated from source-target alignments (b). Symbols in brackets are post-editing operations dictating target word insertion order.

12.1 Introduction

Speech-to-text translation (ST) is an inherently compositional task consisting of first recognizing what was said and then translating the meaning. Cascaded approaches to ST follow this two-staged processing by first performing automatic speech recognition (ASR) and then passing those predictions to a machine translation (MT) model (Waibel, 1996; Inaguma et al., 2020; Lam et al., 2021; Zhang et al., 2022c). On the other hand, end-to-end (E2E) approaches obviate the need for the intermediate ASR task (Bérard et al., 2016; Bahar et al., 2019; Inaguma et al., 2021c; Zhang et al., 2022a), but recent works have demonstrated improved performance using E2E multi-staged architectures which reflect the compositional nature of ST (Sperber et al., 2019; Bahar et al., 2021; Dalmia et al., 2021b; Yan et al., 2022a). However, E2E systems still offer reduced explainability[1] compared to cascaded systems which can use neural MT approaches that explicitly emit word alignment information (Stahlberg et al., 2018).

One direction towards E2E ST explainability is to simultaneously transcribe and translate using the same model. Prior works achieve this by interleaving chunks of ASR and ST predictions within one decoder (Dong et al., 2021; Weller et al., 2021) or by allowing dedicated decoders to model latent alignments between the tasks (Liu et al., 2020a; Le et al., 2020b; Yan et al., 2023a). However, it remains difficult to obtain word-level hard-alignments between ASR and ST sequences from such methods. Ideally, an explainable system would generate a source language word and its aligned target language word in tandem, but this would mean that the resultant translation sequence would be incorrectly ordered. We can prevent generating incorrect word-ordered translations using

[1]In this study, we define explainability as the ability to make the users understand the spoken words are correctly transcribed and translated.

language models, similar to statistical MT (Brown et al., 1990; Koehn et al., 2007a). But it does not resolve this re-ordering problem directly. Recent MT works offer a potential solution via non-monotonic generation methods capable of insertion-based decoding (Gu et al., 2019; Emelianenko et al., 2019; Góis et al., 2020; Ran et al., 2021; Xu and Carpuat, 2021). However, these highly flexible approaches are not directly compatible with the monotonic design of speech processing systems, particularly for real-time streaming (Pham et al., 2019b; Ma et al., 2019a; Arivazhagan et al., 2020; Sperber et al., 2020; Chen et al., 2021; Anastasopoulos et al., 2021; Dong et al., 2022).

In this work, we seek to build explainable E2E ST models which 1) simultaneously generate ASR and ST predictions along with 2) word-level hard-alignment predictions [2] while 3) avoiding degradation to offline and streaming translation quality. In particular, we are interested in modeling *insertions* to allow models to predict ASR tokens monotonically along with their corresponding ST tokens while remembering how to re-order the translation sequence later. We propose to generate a sequence of tuples consisting of a source language word (for ASR), a corresponding target language word (for ST), and post-editing operations dictating the correct insertion point for the target word – we refer to these as *operation sequences* (Stahlberg et al., 2018). In our proposed method, the sequence can be converted into word-level alignments (e.g., Figure 12.1.b), more explainable than outputs of the existing simultaneous ASR/ST models (Dong et al., 2021; Weller et al., 2021; Liu et al., 2020a; Le et al., 2020b; Yan et al., 2023a) (e.g., Figure 12.1.a). Hence, model explainability can be improved by replacing the output with the operation sequence. Our first operation sequence predicts the absolute positions of each target word within the eventual translation sequence (e.g., Figure 12.1.c); however the learned absolute positions may not generalize. To model insertion positions more relatively, we define a second operation sequence that models a shifting write header (e.g., Figure 12.1.d). Our relative shift operation sequence is inspired by (Stahlberg et al., 2018) which was proposed for MT, and we extend this approach to ST by also performing simultaneous prediction of source language words. We demonstrate that our relative shift operation sequence enables explainable and high-quality E2E ST compared to baselines for offline and streaming settings with experiments on MuST-C (Di Gangi et al., 2019). Further, we find that the delayed re-ordering ability of our approach improves the partial hypothesis during streaming. Finally, an added benefit of our method is that simultaneous ASR/ST is faster than using two separate systems.

[2] Conventional MT works show that phrase-level alignment help to predict better translation than word-level alignment (d. Gispert et al., 2004; Koehn et al., 2003). We would extend word-level alignment prediction to phrase-level alignment in the future.

12.2 Background and Motivation

In this section, we review the black-box nature of direct approaches to E2E ST to motivate our explainable E2E ST approach in §12.3.

Direct E2E ST with ASR Multi-Tasking

The objective of the E2E ST is to directly predict a token sequence of the target (t) translation $Y^{(t)} = \{y_l^{(t)} \in \mathcal{Y}^{(t)}\}_{l=1}^{L^{(t)}}$ from an input audio feature sequence $X = \{x_i \in \Re^{D^{(x)}}\}_{i=1}^{I}$. I, $L^{(s)}$, and $L^{(t)}$ denote the sequence length of the input audio feature, transcription, and translation, respectively; $D^{(x)}$ denotes the dimension of input audio feature; and $\mathcal{Y}^{(t)}$ denotes the token vocabulary of target language. To predict translation, a neural network (NN) is trained to maximize the following objective:

$$\begin{aligned}\mathcal{L}^{(\text{ST})} &= \log p(Y^{(t)}|\mathbf{X}) \\ &= \sum_{m=1}^{M} \log p(y_m^{(t)}|y_{1:m-1}^{(t)}, \mathbf{X}),\end{aligned} \tag{12.1}$$

which is frequently modeled using the attention-based encoder-decoder (AED) architecture (Weiss et al., 2017; Karita et al., 2019). AED models trained to maximize Eq. equation 12.1 try to *directly* predict a translation without knowledge of the source language, meaning it has to perform the complex mapping of a continuous speech input to a non-monotonic target language sequence. To alleviate this challenge, we can consider that each audio feature sequence $\mathbf{X}$ has a corresponding token sequence of the source (s) transcription $\mathbf{y}^{(s)} = \{y_l^{(s)} \in \mathcal{Y}^{(s)}\}_{l=1}^{L^{(s)}}$. $\mathcal{Y}^{(s)}$ denotes the token vocabulary of source language. Therefore we apply a multi-tasked ASR and ST loss $\mathcal{L}$ as follows:

$$\begin{aligned}\mathcal{L}^{(\text{ASR})} &= \log p(\mathbf{y}^{(s)}|X) \\ &= \sum_{n=1}^{N} \log p(y_n^{(s)}|y_{1:n-1}^{(s)}, X).\end{aligned} \tag{12.2}$$

$$\mathcal{L} = \alpha\mathcal{L}^{(ST)} + (1-\alpha)\mathcal{L}^{(ASR)}, \tag{12.3}$$

where α denotes a controllable interpolation (we use 0.3 in this study). Using the NN trained with Eq. equation 12.3, we can obtain hypothesized translation token sequence $\hat{y}^{(t)}$ using the following decision:

$$\hat{y}^{(t)} = \underset{y^{(t)} \in \mathcal{Y}^{(t)*}}{\text{argmax}} \log p(y^{(t)}|X), \tag{12.4}$$

where $\mathcal{Y}^{(t)*}$ denotes all possible translation sequences. Typically beam search approximates Eq. equation 12.4, during which the AED generates m-step output token $\hat{y}_m^{(t)}$ in a left-to-right autoregressive manner:

$$\mathbf{h} = \text{Encoder}(X), \tag{12.5}$$

$$\hat{y}_m^{(t)} = \text{Decoder}(\hat{y}_{1:m-1}^{(t)}, \mathbf{h}). \tag{12.6}$$

Similarly, an ASR decoder generates n-step source token $\hat{y}_n^{(s)}$, and ASR targets to feed the multi-tasked loss computation (Eq. equation 12.3):

$$\hat{y}_n^{(s)} = \text{ASRDecoder}(\hat{y}_{1:n-1}^{(s)}, \mathbf{h}), \tag{12.7}$$

where both ST and ASR decoders share the same encoder. Note that ST performance has been shown to benefit from encoder initialization with pre-trained ASR parameters (Eq. equation 12.2) (Bansal et al., 2019).

Shortcomings of the Direct Approach

First and foremost, we are interested in improving the explainability of E2E ST. Considering that the human process for speech-to-text translation entails first recognizing and then translating, the lack of an intermediate ASR output in the direct approach is a major divergence from our natural thought process. Direct E2E ST models with ASR multi-tasking do generate ASR targets during training, but the ASR decoder (Eq. equation 12.7) is parallel to the ST decoder (Eq. equation 12.6) – in other words, ST predictions are not conditioned on any ASR predictions. Therefore, our *first desideratum* is to simultaneously generate ASR and ST predictions such that each ST word may be explained by a corresponding ASR word. Further our objective is not to simply build explainable ST models at all costs, but rather to enhance high-performance models with explainability – this means that we cannot sacrifice translation quality or latency. This latter consideration is particularly important for streaming models. Therefore our *second desideratum* is that our explainable models are as good as direct E2E ST toplines in terms of both translation quality and latency.

12.3 Proposed Framework

We first propose to allow E2E ST models to predict word-aligned ASR and ST sequences simultaneously by formulating a general framework for generating target words out-of-order while remembering how to re-order them via post-editing to obtain final translations. We then present two variants of such operation sequences which model absolute positions or relative shifts for target

tokens to enable the aforementioned delayed re-ordering ability.

Word-Aligned Simultaneous E2E ASR and ST

Suppose that word-level alignments between two languages are known.[3] Let us first re-formulate the ASR and ST targets as word sequences $\bar{y}^{(s)} = \{\bar{y}_i^{(s)}\}_{i=1}^{\bar{L}^{(s)}}$ and $\bar{y}^{(t)} = \{\bar{y}_j^{(t)}\}_{j=1}^{\bar{L}^{(t)}}$, where $\bar{L}^{(s)}$ and $\bar{L}^{(t)}$ denote the number of source and target words. Note that we represent single words, $\bar{y}_i^{(s)}$ or $\bar{y}_j^{(t)}$, as a sequence of sub-word tokens. We can then define word-aligned ASR/ST target sequences consisting of tuples of $(\bar{y}_i^{(s)}, \{\bar{y}_{ik}^{(t)}\}_{k=1}^{\bar{L}_i^{(t)}})$, where each source word $\bar{y}_i^{(s)}$ is aligned to $\bar{L}_i^{(t)}$ target words in a one-to-many mapping. Note that some source or target tokens may actually be un-aligned, so we must augment the respective vocabularies with special tokens [NO_SRC] or [NO_TGT]. Next, consider that simply concatenating all i-steps of target word chunks produces an out-of-order translation if there is any re-ordering between source and target languages (e.g., in Figure 12.1.b), thereby obscuring the intended meaning.

To resolve this out-of-order problem in word-aligned ASR/ST sequences, we propose augmenting each tuple with $\bar{L}_i^{(t)}$ post-editing operations $\{\bar{y}_{ik}^{(o)}\}_{k=1}^{\bar{L}_i^{(t)}}$, obtaining an operation tuple:

$$a_i = (\bar{y}_i^{(s)}, \{\bar{y}_{ik}^{(t)}, \bar{y}_{ik}^{(o)}\}_{k=1}^{\bar{L}_i^{(t)}}). \tag{12.8}$$

These post-editing operations perform rule-based re-ordering of each aligned target word $\bar{y}_{ik}^{(t)}$ by inserting them into correct positions, recovering the original order of the target translation – we discuss particular rule sets to realize this function in the next section.

We can now define an *operation sequence* $A = \{a_i\}_{i=1}^{M}$ and train an AED to predict A by replacing Eq. equation 12.1 with:

$$\begin{aligned} \mathcal{L}^{(OPS)} &= \log p(A|X) \\ &= \sum_{i=1}^{M^\star} \log p(y_i^\star | y_{1:i-1}^\star, X), \end{aligned} \tag{12.9}$$

where $y_i^\star$ denotes elements of the collapsed representation of a_i (i.e., words are collapsed into their token sequences) and $M^\star$ denotes the length of our target sequence. Note that operation sequences are modeled autoregressively by first predicting the source, then the target, and finally the operation (Eq. equation 12.9). On the other hand, the direct E2E ST model trained to maximize Eq. equation 12.3 does not model any explicit relations between its parallel ASR and ST

[3] In this work, we use the statistical tool MGIZA++ (Gao and Vogel, 2008), which performs data-driven alignment without needing additional linguistic resources.

predictions. Further note that we only replace the function of the main decoder Eq. equation 12.6 in our operation sequence AED models, meaning we do not need to sacrifice benefits of ASR multi-tasking (§12.2).

Defining Post-Editing Operations

We examine two ways to represent the insertion position information of post-editing operations: the first uses absolute positions (e.g, Figure 12.1.c) and the second uses relative shifts (e.g., Figure 12.1.d).

Absolute Positional Operation Sequence (Abs-OP)

We define absolute positional sequences (Abs-OP) with Backus-Naur form as:

$< A >$ ::= $< A >< a_i{}^* >$[EOS] | $< a_1{}^* >$[EOS]
$< a_i{}^* >$ ::= $< \bar{y}_i^{(s)} >$[BL]$< T >$
$< T >$ ::= $< T >< \bar{y}_{ik}^{(t)} >< \bar{y}_{ik}^{(o)} >$ | $< \bar{y}_{i1}^{(t)} >< \bar{y}_{i1}^{(o)} >$
$< \bar{y}_i^{(s)} >$ ::= $\bar{y}_i^{(s)}$ | [NO_SRC]
$< \bar{y}_{ik}^{(t)} >$ ::= $\bar{y}_{ik}^{(t)}$ | [NO_TGT]
$< \bar{y}_{ik}^{(o)} >$:= $[n]$ | $[-1]$,

where ::= and | denote definition and choice, respectively; [BL] and [EOS] denote the blank symbol for separating source and target words and the end of the sequence, respectively; $\bar{y}_i^{(s)}$, $\bar{y}_{ik}^{(t)}$ denotes i-th source word in the transcription and k-th target word which corresponds to $\bar{y}_i^{(s)}$; and $[n]$ (where $n \in \mathbb{N}$) denotes the position of $\bar{y}_{ik}^{(t)}$ in the translation. We set n as -1 when $\bar{y}_{ik}^{(t)}$ is [NO_TGT].

To restore the transcription and translations, we prepare sufficient-length buffers for the source transcription $\mathcal{S}$ and the target translation $\mathcal{T}$. Note that all elements of the $\mathcal{S}$ and $\mathcal{T}$ are initialized with [NO_SRC] or [NO_TGT]. During run-time, our model outputs the Abs-OP tokens, and these tokens are inserted into FIFO queue $\mathcal{Q}$. When the model outputs the post-editing command token $< \bar{y}_{ki}^{(o)} >$, we obtain sequences of $< \bar{y}_i^{(s)} >$[BL]$< \bar{y}_i^{(t)} >< \bar{y}_{ki}^{(o)} >$ or $< \bar{y}_i^{(t)} >< \bar{y}_{ki}^{(o)} >$ by dequeuing $\mathcal{Q}$. Then the i-th element of $\mathcal{S}$ is rewritten with $\bar{y}_i^{(s)}$ when the dequeued sequence includes source word. And the $< \bar{y}_{ki}^{(o)} >$-th element of $\mathcal{T}$ is rewritten with $< \bar{y}_{ki}^{(t)} >$. If $< \bar{y}_{ki}^{(o)} >$ indicates -1, we skip this step. This step is repeated until the end of the sequence, and we obtain the source transcription and the target translations from $\mathcal{S}$ and $\mathcal{T}$.

Relative Shift Operation Sequences (Rel-OP)

(Stahlberg et al., 2018) originally proposed an explainable MT model which models insertions

Operations	Transcriptions	Translations
I	I *	*
[NO_OPS]	I *	*
Ich	I *	Ich *
really	I really *	Ich *
[SET_MARKER]	I really *	Ich * *
wirklich	I really *	Ich * wirklich *
need	I really need *	Ich * wirklich *
[JMP_BWD]	I really need *	Ich * wirklich *
brauche	I really need *	Ich brauche * wirklich *
it	I really need it *	Ich brauche * wirklich *
[NO_OPS]	I really need it *	Ich brauche * wirklich *
das	I really need it *	Ich brauche das * wirklich *

Figure 12.2: Restoring transcription and translation from the relative shift operation sequence. [EOP] operations are omitted for space. * denotes a marker and positions of the write-heads are highlighted.

using a moving write header – we adapt this approach for simultaneous ASR/ST. In (Stahlberg et al., 2018), the operation sequence is defined as the stack operations of the read-head $p_{\mathrm{r}}^{(s)}$ for the transcription $\mathcal{S}$ and the write-head $p_{\mathrm{w}}^{(t)}$ for translation $\mathcal{T}$ using following operations:

POP_SRC moves $p_{\mathrm{r}}^{(s)}$ right by one token
SET_MARKER (SM) inserts a marker symbol into $\mathcal{T}$ at $p_{\mathrm{w}}^{(t)}$
JMP_FWD (JF) moves $p_{\mathrm{w}}^{(t)}$ to the closest left maker position
JMP_BWD (JB) moves $p_{\mathrm{w}}^{(t)}$ to the closest right maker position
INSERT(t) inserts a target token t into $\mathcal{T}$ at the position $p_{\mathrm{w}}^{(t)}$

Since we do not know the transcription in advance, we remove $p_{\mathrm{r}}^{(s)}$ and **POP_SRC**. Instead, we define write-head $p_{\mathrm{w}}^{(s)}$ for transcription $\mathcal{S}$. We also revise **INSERT** into as follows:

INSERT-t(t) inserts a target token t into $\mathcal{T}$ at $p_{\mathrm{w}}^{(t)}$
INSERT-s(s) inserts a source token s into $\mathcal{S}$ at $p_{\mathrm{w}}^{(s)}$ and moves $p_{\mathrm{w}}^{(s)}$ right by one token

Our relative shift operation sequences (Rel-OP) are obtained in the same manner as (Stahlberg et al., 2018) and are defined with Backus-Naur form as:

$< A > ::= < A >< a_i{}^* >[\mathrm{EOS}] \mid < a_1{}^* >[\mathrm{EOS}]$
$< a_i{}^* > ::= < \bar{y}_i^{(s)} >< T >[\mathrm{EOP}]$
$< T > ::= < T >< \bar{y}_{ik}^{(o)} >< \bar{y}_{ik}^{(t)} > \mid < \bar{y}_{11}^{(o)} >< \bar{y}_{11}^{(t)} >$

Table 12.1: Comparison of simultaneous E2E ASR/ST models using **absolute position** vs. **relative shift operation** sequences across offline and streaming settings. Better results between these two methods are **bolded** and any simultaneous results which reach/surpass the topline results of single-task E2E models are further **underlined**. ASR and ST performances, as measured by % WER and BLEU, are shown on tst-COMMON / tst-HE of MuST-C English→German and English→French (Di Gangi et al., 2019). Average Lagging (AL) (Ma et al., 2020a) is shown for streaming models.

				EN→DE			EN→FR		
ID	Model	Type	# Params	WER(↓)	BLEU(↑)	AL(↓)	WER(↓)	BLEU(↑)	AL(↓)
A1	Direct E2E ASR (Watanabe et al., 2018) (*Topline*)	Offline	45M	7.7 / 6.7	*no ST*	*n/a*	9.9 / 8.1	*no ST*	*na*
A2	Direct E2E ST (Inaguma et al., 2020) (*Topline*)	Offline	63M	*no ASR*	24.8 / 22.9	*n/a*	*no ASR*	35.9 / 32.9	*n/a*
A3	Direct E2E ASR (Watanabe et al., 2018) (*Topline*)	Streaming	45M	12.0 / 9.9	*no ST*	3365	17.9 / 14.2	*no ST*	3279
A4	Direct E2E ST (Inaguma et al., 2020) (*Topline*)	Streaming	63M	*no ASR*	22.2 / 20.1	5750	*no ASR*	32.1 / 29.5	5648
B1	Simul. E2E ASR/ST w/ Absolute Position (Abs-OP)	Offline	63M	12.1 / 12.0	20.4 / 17.9	*n/a*	14.8 / 16.2	28.7 / 26.0	*n/a*
B2	Simul. E2E ASR/ST w/ Relative Shift (Rel-OP)	Offline	63M	**10.1 / 9.9**	**25.2 / 22.1**	*n/a*	**12.5 / 11.3**	**35.8 / 32.9**	*n/a*
B3	Simul. E2E ASR/ST w/ Absolute Position (Abs-OP)	Streaming	63M	18.7 / 17.2	18.4 / 16.5	5795	25.3 / 23.4	28.7 / 24.5	5684
B4	Simul. E2E ASR/ST w/ Relative Shift (Rel-OP)	Streaming	63M	**16.1 / 14.9**	**23.1 / 20.1**	5786	**19.3 / 16.9**	**32.1 / 29.6**	5770

$< \bar{y}_i^{(s)} >$ ::= $\bar{y}_i^{(s)}$ | [NO_SRC]
$< \bar{y}_{ik}^{(t)} >$ ::= $\bar{y}_{ik}^{(t)}$ | [NO_TGT]
$< \bar{y}_{ik}^{(o)} >$ ::= $< \bar{y}_{ik}^{(o)} >< \bar{y}_{ikj}^{(o)} >$ | $< \bar{y}_{ikj}^{(o)} >$
$< \bar{y}_{ikj}^{(o)} >$ ::= [JF] | [JB] | [SM] | [NO_OPS],

where [EOP] and [NO_OPS] denote the end of sequence for $< \bar{y}_i^{(s)} >$ and no operation is required for $< \bar{y}_{ik}^{(t)} >$, respectively.

Similar to Abs-OP, Rel-OP is inserted into the FIFO queue $\mathcal{Q}$ during run-time. When the model outputs [EOP], we obtain $< a_i^* >$ by dequeuing $\mathcal{Q}$. Then, we can update $\mathcal{S}$ and $\mathcal{T}$ based on the operations. This operation is repeated until the end of the sequence, resulting in transcription and translations from $\mathcal{S}$ and $\mathcal{T}$ by removing the marker symbols. Figure 12.2 depicts how the transcriptions and translations are restored from the proposed Rel-OP.

12.4 Results

Experimental setup

Data: We conduct experiments using the English-German and English-French pairs of the MuST-C corpus (Di Gangi et al., 2019). We apply the tokenizer of the Moses toolkit (Koehn et al., 2007a) for training sentences. Then we remove the samples whose length ratio between the source and target sequences is greater than five or the target sequence length is longer than 150 tokens. To

compute the word alignment, we use MGIZA++ (Gao and Vogel, 2008). For training MGIZA++ model, we set the number of iterations to 5 for HMM, Model 1, Model2, Model 3, and 10 for Model 4; and we use the deficient distortion model for the empty word to reduce the mapping of the target word into [NO_SRC].

Models: Models are trained using ESPnet (Watanabe et al., 2018; Inaguma et al., 2020). We use 4000-vocabulary and 16000-vocabulary BPE (Sennrich et al., 2016) units for source transcription and target operation sequence. We also use 16000-vocabulary BPE units for direct E2E toplines for a fair comparison. ASR and ST models both use conformer encoders (Gulati et al., 2020; Guo et al., 2021) with 12 blocks, 4 heads, 31 kernel size, 2048 feed-forward dim, and 256 attention dim. We initialize the ST encoder with pre-trained ASR parameters for faster convergence. ASR and ST attentional decoders consist of 6 blocks, 4 heads, and 2048 feed-forward dim. Streaming models follow the blockwise method in (Tsunoo et al., 2021) with 40 block size, 16 hop size, and 16 look-ahead. All models are trained for 40 epochs and decoded using 10 beam and 2048 sim chunk length for streaming with repeat detection (Tsunoo et al., 2021).

Evaluation: We evaluate ASR via word error rate (WER↓) and ST via de-tokenized case-insensitive BLEU(↑) (Post, 2018). We evaluate streaming latency via Average Lagging (AL↓) (Ma et al., 2020a); for separate ASR/ST baselines, we consider the latency of both tasks together.

Results and Discussion

Table 12.1 presents our main results on two language pairs across offline and streaming settings. We showed the performance of the direct E2E ASR/ST models to confirm our model output's validity. We found that Rel-OP model produces better translations than Abs-OP model for both offline (`B1` vs. `B2`) and streaming models (`B3` vs. `B4`). The absolute positions appear to inhibit generalization, and we found that these models skewed towards overly short hypotheses even when brevity penalty was applied during beam search. On the other hand, the relative position information from the Rel-OP appears to avoid these pitfalls. Our Rel-OP model achieve comparable performance compared to topline E2E ST models which solely optimize toward translation quality in the offline setting (`A2` vs. `B2`). Impressively, in the streaming setting, Rel-OP model even surpass the performance of dedicated toplines without additional latency (`A4` vs. `B4`), suggesting that the ability to delay re-ordering is particularly useful for streaming ST. This improvement is particularly noticeable in the English-German tst-COMMON set, where Rel-OP model yields 0.9 BLEU gain; we found that this particular set required the most re-ordering between source and target languages.

Since our ST models are also simultaneously predicting ASR predictions, we examine the ASR quality compared to topline models solely focusing on ASR. In Table 12.1, it is clear that the burden of producing explainable ST predictions hinders our operation sequence models from

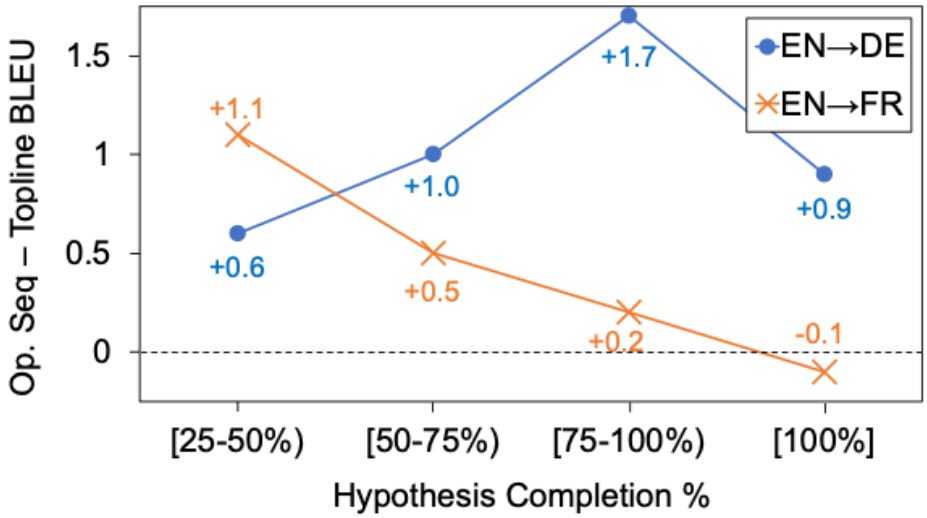

Figure 12.3: Comparison of partially generated ST hypotheses during streaming between relative shift operation sequence models vs. direct ST toplines, measured by absolute BLEU difference on dev sets.

matching the topline ASR models – our models must not only transcribe, but also do so in a way that explains the translations. We also noted that decoding longer translations caused the simultaneous transcriptions to degrade in WER; we do not seek to optimize the ASR side of this trade-off in this work. Nonetheless, the simultaneous transcriptions are reasonable outputs which enable word-level explainability. Further, these transcriptions have no additional latency, so our simultaneous ASR/ST models are faster than two separate dedicated models performing both tasks.

Finally, we are interested in the re-ordering capability of our operational sequence methods during streaming, where systems need to generate translations while still reading the speech signal. We can apply operation sequences *on-the-fly* instead of during post-editing. This allows our systems to produce target words according to the order that they are spoken (monotonically) instead of waiting until all of the preceding target words have been produced (as in direct E2E systems). As shown in Figure 12.3, our Rel-OP models consistently generate better partial hypotheses during streaming than topline direct models. For instance in the English-French case, this advantage is most apparent for very early hypotheses and the gap between the models closes as hypotheses reach their full lengths.

12.5 Conclusion

We propose to build E2E models which simultaneously perform ASR and ST by replacing targets with operation sequences which describe word-level alignments and post-editing commands for re-ordering target language words which were produced out-of-order. Our models using relative shift operation sequences achieve explainability without sacrificing translation quality or achievement. Further, these models actually outperformed direct E2E ST toplines in streaming settings due to an improved word re-ordering ability.

We've now demonstrated static (Chapter 9) and dynamic (Chapter 10) displays for streaming speech translation using sequential, temporal multi-sequence approaches to build models *from scratch*. Next we consider how to enable these types of behaviors in foundation models *via adaptations*.

Chapter 13

Thesis Conclusion

In this thesis, we examined the limitations of the single-sequence paradigm for speech recognition in multilingual settings. While this formulation has driven substantial progress, it fails to capture the ambiguity and variability that arise in real-world speech, particularly under accent variation and code-switching, and fails to support the low-latency required by streaming speech translation applications.

13.1 Summary of Key Contributions

This thesis introduced a multi-sequence perspective for speech recognition and translation in multilingual settings, spanning parallel and sequential formulations under both non-temporal and temporal constraints. The key contributions are summarized as follows.

Multilingual Speech Recognition in the Wild (Parallel, Non-Temporal):

- Identified the limitations of single-sequence modeling as systems scale from tens to hundreds of languages, particularly due to error propagation from language misidentification.
- Proposed Multilingual N-best Re-ranking, an inference-time adaptation that leverages multiple candidate hypotheses to improve robustness in massively multilingual ASR.

Code-Switched Speech Recognition (Parallel, Temporal):

- Systematically evaluated architectures for bilingual and code-switched ASR, highlighting limitations of monolithic modeling approaches.
- Proposed Align-then-Stitch, a framework for composing language-specific predictions into a unified transcription.

- Introduced CS-FLEURS and CS-YODAS, datasets that enable evaluation of code-switching at both controlled and web-scale, in-the-wild settings.
- Scaled code-switched ASR to the massively multilingual regime through the CS-Anything project, extending Align-then-Stitch to support code-switching across 100 languages.

Offline Speech Translation (Sequential, Non-Temporal):

- Evaluated architectures for multilingual and low-resource speech translation, including dialectal settings.
- Proposed Multi-Decoder, an end-to-end differentiable cascade that integrates speech recognition and machine translation within a unified framework.
- Submitted the first place system to the IWSLT 2022 Dialectal Speech Translation Shared Task for Tunisian Arabic to English speech-to-text translation

Streaming Speech Translation (Sequential, Temporal):

- Evaluated architectures for speech translation under real-time, low-latency constraints.
- Proposed Hierarchical CTC and Post-edit Operations to improve streaming performance.
- Submitted the first place system to the IWSLT 2023 Simulateneous Speech Translation Shared Task for English to German speech-to-speech translation

13.2 Future Directions

This thesis focused on improving systems in two general areas: we wanted systems to support more language variety and we wanted systems to perform with super low latency. The broader impact of this line of thinking is **building systems which support language in motion**.

Language is dynamic and fluid, as it is used by populations of people who are themselves in motion. Motion can happen gradually over time within communities, taking place as new words are invented or borrowed. Motion can also be quick interactions between individuals, taking place as random conversations between strangers who speak different languages.

The current generation of speech systems can be deployed, as we have shown, to support instances of language in motion such as regional code-switching and speech-to-speech translation. In future work, there are two major opportunities to expand the frontier of our systems:

- **New Code-Switching:** Code-switching is constantly evolving, driven by interactions across cultural, societal, and interpersonal contexts. As these dynamics shift, so do the patterns of code-switching. This raises a key challenge: how well can current systems handle out-of-distribution code-switching, and how quickly and effectively can they adapt to newly emerging or borrowed terms? For instance, English machine learning terminology has risen in global popularity in recent years. Are we able to recognize a Hindi speaker using those code-switched or borrowed terms even if that never occurred in our training data? These ideas are a bridge towards the related field of contextual biasing, but in code-switching the presence of multiple languages opens up a number of challenges such as accent variations and cross-lingual confusions.
- **World Translation:** Real-time, face-to-face, speech-to-speech translation systems work and are already in practical use, but they are still limited. Current speech translation systems are designed to translate a single speaker and can do so in the presence of noise and other background speakers. But what if you wanted to translate beyond a single speaker? Let's say you are a foreign exchange student and you walked into an auditorium of people in various research conversations – could you walk around and understand what everyone is talking about? Ultimately, this would mean running speech translation in an ambient mode and doing so with low latency would be exceptionally challenging given the increased complexity of the system; however, if successful this line of work would go a long way towards enabling seamless global communication.

Bibliography

Oliver Adams, Matthew Wiesner, Shinji Watanabe, and David Yarowsky. 2019. Massively multilingual adversarial speech recognition. In *Proc. NAACL-HLT*.

Milind Agarwal, Sweta Agrawal, Antonios Anastasopoulos, Ondřej Bojar, Claudia Borg, Marine Carpuat, Roldano Cattoni, Mauro Cettolo, Mingda Chen, William Chen, Khalid Choukri, Alexandra Chronopoulou, Anna Currey, Thierry Declerck, Qianqian Dong, Yannick Estève, Kevin Duh, Marcello Federico, Souhir Gahbiche, Barry Haddow, Benjamin Hsu, Phu Mon Htut, Hirofumi Inaguma, Dávid Javorský, John Judge, Yasumasa Kano, Tom Ko, Rishu Kumar, Pengwei Li, Xutail Ma, Prashant Mathur, Evgeny Matusov, Paul McNamee, John P. McCrae, Kenton Murray, Maria Nadejde, Satoshi Nakamura, Matteo Negri, Ha Nguyen, Jan Niehues, Xing Niu, Atul Ojha Kr., John E. Ortega, Proyag Pal, Juan Pino, Lonneke van der Plas, Peter Polák, Elijah Rippeth, Elizabeth Salesky, Jiatong Shi, Matthias Sperber, Sebastian Stüker, Katsuhito Sudoh, Yun Tang, Brian Thompson, Kevin Tran, Marco Turchi, Alex Waibel, Mingxuan Wang, Shinji Watanabe, and Rodolfo Zevallos. 2023. Findings of the IWSLT 2023 Evaluation Campaign. In *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*. Association for Computational Linguistics.

Emily Ahn, Cecilia Jimenez, Yulia Tsvetkov, and Alan W Black. 2020. What code-switching strategies are effective in dialog systems? In *Proc. SCiL*.

Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A Smith, and David Yarowsky. 1999. Statistical machine translation. In *Final Report, JHU Summer Workshop*, volume 30.

Ahmed Ali, Peter Bell, James Glass, Yacine Messaoui, Hamdy Mubarak, Steve Renals, and Yifan Zhang. 2016. The mgb-2 challenge: Arabic multi-dialect broadcast media recognition. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 279–284. IEEE.

Ahmed M. Ali, Shammur A. Chowdhury, A. Hussein, and Yasser Hifny. 2021. Arabic code-switching speech recognition using monolingual data. In *Interspeech*.

A. Anastasopoulos, O. Bojar, J. Bremerman, et al. 2021. Findings of the IWSLT 2021 evaluation campaign. In *Proc. of IWSLT*, pages 1–29.

Antonios Anastasopoulos, Luisa Bentivogli, Marcely Z. Boito, Ondřej Bojar, Roldano Cattoni, Anna Currey, Georgiana Dinu, Kevin Duh, Maha Elbayad, Marcello Federico, Christian Federmann, Hongyu Gong, Roman Grundkiewicz, Barry Haddow, Benjamin Hsu, Dávid Javorský, Věra Kloudová, Surafel M. Lakew, Xutai Ma, Prashant Mathur, Paul McNamee, Kenton Murray, Maria Nădejde, Satoshi Nakamura, Matteo Negri, Jan Niehues, Xing Niu, Juan Pino, Elizabeth Salesky, Jiatong Shi, Sebastian Stüker, Katsuhito Sudoh, Marco Turchi, Yogesh Virkar, Alex Waibel, Changhan Wang, and Shinji Watanabe. 2022. FINDINGS OF THE IWSLT 2022 EVALUATION CAMPAIGN. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, Dublin, Ireland. Association for Computational Linguistics.

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 82–91, New Orleans, Louisiana. Association for Computational Linguistics.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 39–48. IEEE Computer Society.

Alan Ansell, Edoardo Ponti, Anna Korhonen, and Ivan Vulić. 2022. Composable sparse fine-tuning for cross-lingual transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1778–1796.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. 2020. Common voice: A massively-multilingual speech corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4218–4222.

N. Arivazhagan, C. Cherry, I Te, et al. 2020. Re-translation strategies for long form, simultaneous, spoken language translation. In *Proc. of ICASSP*, pages 7919–7923.

Peter Auer. 2013. *Code-switching in conversation: Language, interaction and identity*. Routledge.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

P. Bahar, T. Bieschke, and H. Ney. 2019. A comparative study on end-to-end speech to text translation. In *Proc. ASRU 2019*, pages 792–799.

Parnia Bahar, Tobias Bieschke, Ralf Schlüter, and Hermann Ney. 2021. Tight integrated end-to-end training for cascaded speech translation. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 950–957. IEEE.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Junwen Bai, Bo Li, Yu Zhang, Ankur Bapna, Nikhil Siddhartha, Khe Chai Sim, and Tara N. Sainath. 2022. Joint unsupervised and supervised training for multilingual asr. In *ICASSP*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

S. Bansal, H. Kamper, K. Livescu, et al. 2019. Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. In *Proc. NAACL*, pages 58–68.

Ankur Bapna, Colin Cherry, Yu Zhang, Ye Jia, Melvin Johnson, Yong Cheng, Simran Khanuja, Jason Riesa, and Alexis Conneau. 2022. mslam: Massively multilingual joint pre-training for speech and text. *arXiv preprint arXiv:2202.01374*.

Daniel Beck, Trevor Cohn, and Gholamreza Haffari. 2019. Neural speech translation using lattice transformations and graph networks. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 26–31, Hong Kong. Association for Computational Linguistics.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2020. On the linguistic representational power of neural machine translation models. *Computational Linguistics*, 46(1):1–52.

A. Bérard, O. Pietquin, C. Servan, and L. Besacier. 2016. Listen and translate: A proof of concept for end-to-end speech-to-text translation. *arXiv preprint arXiv:1612.01744*.

Jayadev Billa. 2021. Leveraging Non-Target Language Resources to Improve ASR Performance in a Target Language. In *Interspeech*.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Léon Bottou. 1991. *Une Approche théorique de l'Apprentissage Connexionniste: Applications à la Reconnaissance de la Parole*. Ph.D. thesis, Université de Paris XI, Orsay, France.

Léon Bottou, Yoshua Bengio, and Yann Le Cun. 1997. Global training of doc processing systems using graph transformer networks. In *CVPR*.

Hervé Bredin. 2023. pyannote.audio 2.1 speaker diarization pipeline: principle, benchmark, and recipe. In *Proc. INTERSPEECH 2023*.

P. F. Brown, J. Cocke, P. Della, et al. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng. 2017. Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline. In *O-COCOSDA*.

Laurie Burchell, Alexandra Birch, Robert P Thompson, and Kenneth Heafield. 2024. Code-switched language identification is harder than you think. *arXiv preprint arXiv:2402.01505*.

Edresson Casanova, Kelly Davis, Eren Gölge, Görkem Göknar, Iulian Gulea, Logan Hart, Aya Aljafari, Joshua Meyer, Reuben Morais, Samuel Olayemi, et al. 2024. Xtts: a massively multilingual zero-shot text-to-speech model. *arXiv preprint arXiv:2406.04904*.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit3: Web inventory of transcribed and translated talks. In *Conference of european association for machine translation*, pages 261–268.

Joyce YC Chan, PC Ching, Tan Lee, and Helen M Meng. 2004. Detection of language boundary in code-switching by bi-phone probabilities. In *Proc. ISCSLP*.

William Chan, Navdeep Jaitly, Quoc V Le, and Oriol Vinyals. 2015. Listen, attend and spell. *arXiv preprint arXiv:1508.01211*.

William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *ICASSP*.

William Chan, Chitwan Saharia, Geoffrey Hinton, Mohammad Norouzi, and Navdeep Jaitly. 2020. Imputer: Sequence modelling via imputation and dynamic programming. In *International Conference on Machine Learning*, pages 1403–1413. PMLR.

Chander Chandak, Zeynab Raeesy, Ariya Rastrow, Yuzong Liu, Xiangyang Huang, Siyu Wang, Dong Kwon Joo, and Roland Maas. 2020. Streaming language identification using combination of acoustic representations and asr hypotheses. *arXiv preprint arXiv:2006.00703*.

Ching-Ting Chang, Shun-Po Chuang, and Hung-Yi Lee. 2018. Code-switching sentence generation by generative adversarial networks and its application to data augmentation. *arXiv preprint arXiv:1811.02356*.

Xuankai Chang, Takashi Maekaku, Pengcheng Guo, Jing Shi, Yen-Ju Lu, Aswin Shanmugam Subramanian, Tianzi Wang, Shu-wen Yang, Yu Tsao, Hung-yi Lee, et al. 2021. An exploration of self-supervised pretrained representations for end-to-end speech recognition. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 228–235. IEEE.

J. Chen, M Ma, R. Zheng, and L. Huang. 2021. Direct simultaneous speech-to-text translation assisted by synchronized streaming ASR. In *Proc. of ACL-IJCNLP*, pages 4618–4624.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. 2022. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518.

William Chen, Brian Yan, Jiatong Shi, Yifan Peng, Soumi Maiti, and Shinji Watanabe. 2023. Improving massively multilingual asr with auxiliary ctc objectives. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. *Advances in neural information processing systems*, 28.

Shammur Absar Chowdhury, Amir Hussein, Ahmed Abdelali, and Ahmed Ali. 2021. Towards one model to rule all: Multilingual strategy for dialectal code-switching arabic asr. In *Interspeech 2021*, pages 2466–2470.

Shun-Po Chuang, Yung-Sung Chuang, Chih-Chiang Chang, and Hung-yi Lee. 2021. Investigating the reordering capability in CTC-based non-autoregressive end-to-end speech translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1068–1077, Online. Association for Computational Linguistics.

Shun-Po Chuang, Tzu-Wei Sung, and Hung-yi Lee. 2020. Training code-switching language model with monolingual data. In *Proc. ICASSP*.

Seamless Communication, Loïc Barrault, Yu-An Chung, Mariano Cora Meglioli, David Dale, Ning Dong, et al. 2023. Seamlessm4t: Massively multilingual & multimodal machine translation. *arXiv*.

Alexis Conneau, Min Ma, Simran Khanuja, Yu Zhang, Vera Axelrod, Siddharth Dalmia, Jason Riesa, Clara Rivera, and Ankur Bapna. 2023. Fleurs: Few-shot learning evaluation of universal representations of speech. In *Proc. of SLT*.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. In *Privacy in Machine Learning and Artificial Intelligence workshop, ICML*.

A. d. Gispert, J. B. Marino, and J. M. Crego. 2004. Phrase-based alignment combining corpus cooccurrences and linguistic knowledge. In *Proc. IWSLT*.

Siddharth Dalmia, Yuzong Liu, Srikanth Ronanki, and Katrin Kirchhoff. 2021a. Transformer-transducers for code-switched speech recognition. In *Proc. ICASSP*.

Siddharth Dalmia, Abdelrahman Mohamed, Mike Lewis, Florian Metze, and Luke Zettlemoyer. 2019. Enforcing encoder-decoder modularity in sequence-to-sequence models. *arXiv*.

Siddharth Dalmia, Dmytro Okhonko, Mike Lewis, Sergey Edunov, Shinji Watanabe, Florian Metze, Luke Zettlemoyer, and Abdelrahman Mohamed. 2022. LegoNN: Building modular encoder-decoder models. *arXiv preprint arXiv:2206.03318*.

Siddharth Dalmia, Brian Yan, Vikas Raunak, Florian Metze, and Shinji Watanabe. 2021b. Searchable hidden intermediates for end-to-end models of decomposable sequence tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1882–1896, Online. Association for Computational Linguistics.

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. 2019. ArcFace: Additive angular margin loss for deep face recognition. In *Proc. CVPR*.

Keqi Deng, Shinji Watanabe, Jiatong Shi, and Siddhant Arora. 2022a. Blockwise streaming transformer for spoken language understanding and simultaneous speech translation. *arXiv preprint arXiv:2204.08920*.

Shuhao Deng, Chengfei Li, Qingqing Zhang, Wei-Qiang Zhang, Runyan Yang, Gaofeng Cheng, Pengyuan Zhang, Yonghong Yan, et al. 2022b. Summary on the iscslp 2022 chinese-english code-switching asr challenge. *arXiv preprint arXiv: 2210.06091*.

Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck. 2020. ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification. In *Proc. Interspeech*, pages 3830–3834.

Margaret Deuchar, Peredur Davies, Jon Herring, M Carmen Parafita Couto, and Diana Carter. 2014. Building bilingual corpora. *Advances in the Study of Bilingualism*, pages 93–111.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings NAACL*, pages 4171–4186. Association for Computational Linguistics.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. MuST-C: a Multilingual Speech Translation Corpus. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2012–2017, Minneapolis, Minnesota. Association for Computational Linguistics.

Anuj Diwan, Rakesh Vaideeswaran, Sanket Shah, Ankita Singh, Srinivasa Raghavan, Shreya Khare, Vinit Unni, Saurabh Vyas, Akash Rajpuria, Chiranjeevi Yarra, Ashish Mittal, Prasanta Kumar Ghosh, Preethi Jyothi, Kalika Bali, Vivek Seshadri, Sunayana Sitaram, Samarth Bharadwaj, Jai Nanavati, Raoul Nanavati, and Karthik Sankaranarayanan. 2021a. Mucs 2021: Multilingual and code-switching asr challenges for low resource indian languages. In *Interspeech 2021*, pages 2446–2450.

Anuj Diwan, Rakesh Vaideeswaran, Sanket Shah, Ankita Singh, Srinivasa Raghavan, Shreya Khare, Vinit Unni, Saurabh Vyas, Akash Rajpuria, Chiranjeevi Yarra, et al. 2021b. Multilingual and code-switching asr challenges for low resource indian languages. *arXiv preprint arXiv:2104.00235*.

Linhao Dong, Shuang Xu, and Bo Xu. 2018. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888. IEEE.

Q. Dong, M. Wang, H. Zhou, et al. 2021. Consecutive decoding for speech-to-text translation. In *Proc. AAAI*.

Q. Dong, Y. Zhu, M. Wang, and L. Li. 2022. Learning when to translate for streaming speech. In *Proc. ACL*, pages 680–694.

Qianqian Dong, Mingxuan Wang, Hao Zhou, Shuang Xu, Bo Xu, and Lei Li. 2020. SDST: Successive decoding for speech-to-text translation. *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*.

Zi-Yi Dou and Graham Neubig. 2021. Word alignment by fine-tuning embeddings on parallel corpora. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128.

Marko Dragojevic, Jessica Gasiorek, and Howard Giles. 2015. Communication accommodation theory. *Intl. Encyclopedia of Interpersonal Comm.*

Yichao Du, Zhirui Zhang, Weizhi Wang, Boxing Chen, Jun Xie, and Tong Xu. 2022. Regularizing end-to-end speech translation with triangular decomposition agreement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10590–10598.

Bryan Eikema and Wilker Aziz. 2020. Is MAP decoding all you need? the inadequacy of the mode in neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4506–4520, Barcelona, Spain (Online). International Committee on Computational Linguistics.

D. Emelianenko, E. Voita, and P. Serdyukov. 2019. Sequence modeling with unconstrained generation order. *Advances in Neural Information Processing Systems*, 32.

J.G. Fiscus. 1997. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover). In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pages 347–354.

Marco Gaido, Mauro Cettolo, Matteo Negri, and Marco Turchi. 2021. CTC-based compression for direct speech translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 690–696.

Björn Gambäck and Amitava Das. 2014. Comparing the level of code-switching in corpora. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC)*, pages 1850–1855, Reykjavik, Iceland.

Q. Gao and S. Vogel. 2008. Parallel implementations of word alignment tool. In *Proc. SETQA-NLP*, pages 49–57, Columbus, Ohio.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *International conference on machine learning*, pages 1243–1252. PMLR.

Xinwei Geng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2018. Adaptive multi-pass decoder for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 523–532, Brussels, Belgium. Association for Computational Linguistics.

Marjan Ghazvininejad, Vladimir Karpukhin, Luke Zettlemoyer, and Omer Levy. 2020. Aligned cross entropy for non-autoregressive machine translation. In *International Conference on Machine Learning*, pages 3515–3523. PMLR.

John J Godfrey, Edward C Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, volume 1, pages 517–520. IEEE Computer Society.

A. Góis, K. Cho, and A. Martins. 2020. Learning non-monotonic automatic post-editing of translations from human orderings. *arXiv preprint arXiv:2004.14120*.

Hila Gonen and Yoav Goldberg. 2018. Language modeling for code-switching: Evaluation, integration of monolingual data, and discriminative training. *Proc. EMNLP*.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2022. The flores-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538.

A. Graves. 2012a. Sequence Transduction with Recurrent Neural Networks. In *Proc. ICML*.

Alex Graves. 2012b. Connectionist temporal classification. In *Supervised sequence labelling with recurrent neural networks*, pages 61–93. Springer.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proc. ICML*.

Alex Graves and Alex Graves. 2012. *Supervised sequence labelling*. Springer.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610. IJCNN 2005.

J. Gu, Q. Liu, and K. Cho. 2019. Insertion-based decoding with automatically inferred generation order. *Trans. of the ACL*, 7:661–676.

Jiatao Gu and Xiang Kong. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 120–133.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. 2020. Conformer: Convolution-augmented transformer for speech recognition. *Interspeech*.

Pengcheng Guo, Florian Boyer, Xuankai Chang, Tomoki Hayashi, Yosuke Higuchi, Hirofumi Inaguma, Naoyuki Kamo, Chenda Li, Daniel Garcia-Romero, Jiatong Shi, et al. 2021. Recent developments on espnet toolkit boosted by conformer. In *Proc. ICASSP*.

Nizar Habash, Abdelhadi Soudi, and Timothy Buckwalter. 2007. On arabic transliteration. In *Arabic computational morphology*, pages 15–22. Springer.

Parisa Haghani, Arun Narayanan, Michiel Bacchiani, Galen Chuang, Neeraj Gaur, Pedro Moreno, Rohit Prabhavalkar, Zhongdi Qu, and Austin Waters. 2018. From audio to semantics: Approaches to end-to-end spoken language understanding. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 720–726. IEEE.

Injy Hamed, Nizar Habash, Slim Abdennadher, and Ngoc Thang Vu. 2022. ArzEn-ST: A three-way speech translation corpus for code-switched Egyptian Arabic-English. In *Proc WANLP*, pages 119–130.

Injy Hamed, Nizar Habash, Slim Abdennadher, and Ngoc Thang Vu. 2023. Investigating lexical replacements for arabic-english code-switched data augmentation. In *Proc LoResMT*, pages 86–100.

Awni Hannun. 2019. The Label Bias Problem. `https://awni.github.io/label-bias`.

Awni Hannun, Vineel Pratap, Jacob Kahn, and Wei-Ning Hsu. 2020. Differentiable weighted finite-state transducers. *arXiv preprint arXiv:2010.01003*.

Awni Y Hannun, Andrew L Maas, Daniel Jurafsky, and Andrew Y Ng. 2014. First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns. *arXiv preprint arXiv:1408.2873*.

Adi Haviv, Lior Vassertail, and Omer Levy. 2021. Can latent alignments improve autoregressive machine translation? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2637–2641, Online. Association for Computational Linguistics.

Xiaodong He and Li Deng. 2011. Speech recognition, machine translation, and speech translation—a unified discriminative learning paradigm [lecture notes]. *IEEE Signal Processing Magazine*, 28(5):126–133.

Jindřich Helcl, Jindřich Libovický, and Dušan Variš. 2018. CUNI system for the WMT18 multimodal translation task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 616–623, Belgium, Brussels. Association for Computational Linguistics.

Roberto R Heredia and Jeanette Altarriba. 2001. Bilingual language mixing: Why do bilinguals code-switch? *Curr. Dir. in Psychological Science*.

Ulf Hermjakob, Jonathan May, and Kevin Knight. 2018. Out-of-the-box universal romanization tool uroman. In *Proc. of ACL*.

Yosuke Higuchi, Keita Karube, Tetsuji Ogawa, and Tetsunori Kobayashi. 2022. Hierarchical conditional end-to-end ASR with CTC and multi-granular subword units. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7797–7801. IEEE.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *CoRR*, abs/1904.09751.

Takaaki Hori, Shinji Watanabe, and John Hershey. 2017a. Joint CTC/attention decoding for end-to-end speech recognition. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 518–529, Vancouver, Canada. Association for Computational Linguistics.

Takaaki Hori, Shinji Watanabe, Yu Zhang, and William Chan. 2017b. Advances in joint CTC-attention based end-to-end speech recognition with a deep CNN encoder and RNN-LM. In *Proc. Interspeech 2017*, pages 949–953.

Leijing Hou, Ying Liu, Yingying Gao, and Junlan Feng. 2020a. A study of types and characteristics of code-switching in mandarin-english speech. *Proc. WSTCSMC*.

Wenxin Hou, Yue Dong, Bairong Zhuang, Longfei Yang, Jiatong Shi, and Takahiro Shinozaki. 2020b. Large-scale end-to-end multilingual speech recognition and language identification with multi-task learning. In *Interspeech*, pages 1037–1041.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460.

Ke Hu, Tara N Sainath, Ruoming Pang, and Rohit Prabhavalkar. 2020. Deliberation model based two-pass end-to-end speech recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7799–7803. IEEE.

Chenyang Huang, Hao Zhou, Osmar R Zaïane, Lili Mou, and Lei Li. 2022. Non-autoregressive translation with layer-wise prediction and deep supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10776–10784.

Liang Huang and David Chiang. 2007. Forest Rescoring: Faster decoding with integrated language models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151, Prague, Czech Republic. Association for Computational Linguistics.

Xuedong Huang, James Baker, and Raj Reddy. 2014. A historical perspective of speech recognition. *Communications of the ACM*, 57(1):94–103.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Amir Hussein, Desh Raj, Matthew Wiesner, Daniel Povey, Paola Garcia, and Sanjeev Khudanpur. 2024a. Enhancing neural transducer for multilingual asr with synchronized language diarization. In *INTERSPEECH*, pages 3994–3998.

Amir Hussein, Shinji Watanabe, and Ahmed Ali. 2022. Arabic speech recognition by end-to-end, modular systems and human. *Computer Speech & Language*, 71:101272.

Amir Hussein, Dorsa Zeinali, Ondřej Klejch, Matthew Wiesner, Brian Yan, Shammur Chowdhury, Ahmed Ali, Shinji Watanabe, and Sanjeev Khudanpur. 2024b. Speech collage: code-switched audio generation by collaging monolingual corpora. In *Proc. ICASSP*, pages 12006–12010.

Hirofumi Inaguma, Siddharth Dalmia, Brian Yan, and Shinji Watanabe. 2021a. Fast-MD: Fast multi-decoder end-to-end speech translation with non-autoregressive hidden intermediates. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 922–929. IEEE.

Hirofumi Inaguma, Yosuke Higuchi, Kevin Duh, Tatsuya Kawahara, and Shinji Watanabe. 2021b. Non-autoregressive end-to-end speech translation with parallel autoregressive rescoring. *arXiv preprint arXiv:2109.04411*.

Hirofumi Inaguma, Yosuke Higuchi, Kevin Duh, Tatsuya Kawahara, and Shinji Watanabe. 2021c. Orthros: Non-autoregressive end-to-end speech translation with dual-decoder. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7503–7507. IEEE.

Hirofumi Inaguma, Shun Kiyono, Kevin Duh, Shigeki Karita, Nelson Yalta, Tomoki Hayashi, and Shinji Watanabe. 2020. ESPnet-ST: All-in-one speech translation toolkit. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 302–311, Online. Association for Computational Linguistics.

Javier Iranzo-Sánchez, Joan Albert Silvestre-Cerda, Javier Jorge, Nahuel Roselló, Adria Giménez, Albert Sanchis, Jorge Civera, and Alfons Juan. 2020. Europarl-ST: A multilingual corpus for speech translation of parliamentary debates. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8229–8233. IEEE.

Fei Jia, Nithin Rao Koluguri, Jagadeesh Balam, and Boris Ginsburg. 2023. A compact end-to-end model with local and global context for spoken language identification. In *Proc. Interspeech*, pages 5321–5325.

Patricia Johnson. 1992. Cohesion and coherence in compositions in Malay and English. *RELC Journal*, 23(2):1–17.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomáš Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th conference of the European chapter of the association for computational linguistics: volume 2, short papers*, pages 427–431.

Marcin Junczys-Dowmunt. 2018. Dual conditional cross-entropy filtering of noisy parallel corpora. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 888–895.

Jee-weon Jung, Wangyou Zhang, Jiatong Shi, Zakaria Aldeneh, Takuya Higuchi, Barry-John Theobald, Ahmed Hussen Abdelaziz, and Shinji Watanabe. 2024. ESPnet-SPK: Full pipeline

speaker embedding toolkit with reproducible recipes, self-supervised front-ends, and off-the-shelf models. *arXiv preprint arXiv:2401.17230*.

Preethi Jyothi and Mark Hasegawa-Johnson. 2015. Transcribing continuous speech using mismatched crowdsourcing. In *Interspeech*.

Jacob Kahn, Morgane Riviere, Weiyi Zheng, Evgeny Kharitonov, Qiantong Xu, Pierre-Emmanuel Mazaré, Julien Karadayi, Vitaliy Liptchinsky, Ronan Collobert, Christian Fuegen, et al. 2020. Libri-light: A benchmark for asr with limited or no supervision. In *Proc ICASSP*, pages 7669–7673. IEEE.

S. Karita, N. Chen, T. Hayashi, et al. 2019. A comparative study on transformer vs RNN in speech applications. In *Proc. ASRU*, pages 449–456.

Jaehyeon Kim, Jungil Kong, and Juhee Son. 2021. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*. PMLR.

Suyoun Kim, Takaaki Hori, and Shinji Watanabe. 2017. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4835–4839. IEEE.

Suyoun Kim and Michael L Seltzer. 2018. Towards language-universal end-to-end speech recognition. In *Proc. ICASSP*.

Yoon Kim and Alexander M. Rush. 2016a. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Yoon Kim and Alexander M Rush. 2016b. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015*.

Kevin Knight and Jonathan Graehl. 1998. Machine transliteration. *Computational Linguistics*.

Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. 2015. Audio augmentation for speech recognition. In *Proceedings of Interspeech*, pages 3586–3589.

P. Koehn, H. Hoang, A. Birch, et al. 2007a. Moses: Open source toolkit for statistical machine translation. In *Proc. ACL*, pages 177–180.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007b. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133.

Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the second workshop on statistical machine translation*, pages 224–227.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. 2020. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. In *Advances in Neural Information Processing Systems*, volume 33, pages 17022–17033.

Taku Kudo and John Richardson. 2018a. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Taku Kudo and John Richardson. 2018b. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Gaurav Kumar, Matt Post, Daniel Povey, and Sanjeev Khudanpur. 2014. Some insights from translating conversational telephone speech. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, pages 3231–3235. IEEE.

Shankar Kumar and William Byrne. 2002. Minimum bayes-risk word alignments of bilingual texts. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, page 140–147, USA. Association for Computational Linguistics.

Shankar Kumar and William Byrne. 2004. Minimum Bayes-risk decoding for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 169–176, Boston, Massachusetts, USA. Association for Computational Linguistics.

Shankar Kumar, Yonggang Deng, and William Byrne. 2006. A weighted finite state transducer translation template model for statistical machine translation. *Natural Language Engineering*, 12(1):35–76.

Chih-Hua Kuo. 1995. Cohesion and coherence in academic writing: From lexical choice to organization. *RELC Journal*, 26(1):47–62.

Ludwig Kürzinger, Dominik Winkelbauer, Lujun Li, Tobias Watzel, and Gerhard Rigoll. 2020. CTC-segmentation of large corpora for german end-to-end speech recognition. In *SPECOM*.

Garry Kuwanto, Chaitanya Agarwal, Genta Indra Winata, and Derry Tanti Wijaya. 2024. Linguistics theory meets llm: Code-switched text generation via equivalence constrained large language models. *arXiv preprint arXiv:2410.22660*.

Brenden M Lake. 2019. Compositional generalization through meta sequence-to-sequence learning. In *Advances in Neural Information Processing Systems*, pages 9791–9801.

Brenden M. Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, pages 2879–2888. PMLR.

T. K. Lam, S. Schamoni, and S. Riezler. 2021. Cascaded models with cyclic feedback for direct speech translation. In *Proc. ICASSP 2021*, pages 7508–7512.

Hang Le, Juan Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. 2020a. Dual-decoder transformer for joint automatic speech recognition and multilingual speech translation. *Proceedings of the 28th International Conference on Computational Linguistics*.

Hang Le, Juan Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. 2020b. Dual-decoder transformer for joint automatic speech recognition and multilingual speech translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3520–3533.

Grandee Lee, Xianghu Yue, and Haizhou Li. 2019. Linguistically motivated parallel data augment for code-switch language modeling. In *Interspeech*.

Jaesong Lee and Shinji Watanabe. 2021. Intermediate loss regularization for ctc-based speech recognition. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6224–6228.

Alexander H. Levis, Neville Moray, and Baosheng Hu. 1994. Task decomposition and allocation problems and discrete event systems. *Automatica*, 30(2):203 – 216.

M Paul Lewis. 2009. *Ethnologue: Languages of the world*. SIL international.

Bo Li, Ruoming Pang, Tara N Sainath, Anmol Gulati, Yu Zhang, James Qin, Parisa Haghani, W Ronny Huang, Min Ma, and Junwen Bai. 2021. Scaling end-to-end models for large-scale multilingual asr. In *ASRU*.

Bo Li, Tara N. Sainath, Khe Chai Sim, Michiel Bacchiani, Eugene Weinstein, Patrick Nguyen, Zhifeng Chen, Yanghui Wu, and Kanishka Rao. 2018. Multi-dialect speech recognition with a single sequence-to-sequence model. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4749–4753.

Bo Li, Yu Zhang, Tara Sainath, Yonghui Wu, and William Chan. 2019a. Bytes are all you need: End-to-end multilingual speech recognition with bytes. In *Proc. ICASSP*.

Jinyu Li et al. 2022a. Recent advances in end-to-end automatic speech recognition. *APSIPA Transactions on Signal and Information Processing*, 11(1).

Ke Li, Jinyu Li, Guoli Ye, Rui Zhao, and Yifan Gong. 2019b. Towards code-switching asr for end-to-end ctc models. In *Proc. ICASSP*.

Shuyue Stella Li, Cihan Xiao, Tianjian Li, and Bismarck Odoom. 2023a. Simple yet effective code-switching language identification with multitask pre-training and transfer learning. *arXiv preprint arXiv:2305.19759*.

Xian Li, Changhan Wang, Yun Tang, Chau Tran, Yuqing Tang, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. 2020. Multilingual speech translation with efficient finetuning of pretrained models. *arXiv preprint arXiv:2010.12829*.

Xinjian Li, Florian Metze, David R Mortensen, Alan W Black, and Shinji Watanabe. 2022b. Asr2k: Speech recognition for around 2000 languages without audio. *Interspeech*.

Xinjian Li, Shinnosuke Takamichi, Takaaki Saeki, William Chen, Sayaka Shiota, and Shinji Watanabe. 2023b. Yodas: Youtube-oriented dataset for audio and speech. In *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–8. IEEE.

Jindřich Libovický and Jindřich Helcl. 2018. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3016–3021, Brussels, Belgium. Association for Computational Linguistics.

Peiqin Lin, Shaoxiong Ji, Jörg Tiedemann, André FT Martins, and Hinrich Schütze. 2024. Mala-500: Massive language adaptation of large language models. *arXiv preprint arXiv:2401.13303*.

Guoyu Liu and Lixin Cao. 2021. Code-switch speech rescoring with monolingual data. In *ICASSP*.

Hexin Liu, Leibny Paola Garcia-Perera, Xinyi Zhang, Justin Dauwels, Andy WH Khong, Sanjeev Khudanpur, and Suzy J Styles. 2021. End-to-end language diarization for bilingual code-switching speech. In *Interspeech*.

Hexin Liu, Haihua Xu, Leibny Paola Garcia, Andy WH Khong, Yi He, and Sanjeev Khudanpur. 2023. Reducing language confusion for code-switching speech recognition with token-level language diarization. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Hexin Liu, Xiangyu Zhang, Haoyang Zhang, Leibny Paola Garcia-Perera, Andy WH Khong, Eng Siong Chng, and Shinji Watanabe. 2025. Aligning speech to languages to enhance code-switching speech recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 33:4712–4725.

Hong Liu. 2019. Attitudes toward different types of chinese-english code-switching. *Sage Open*.

Y. Liu, J. Zhang, H. Xiong, et al. 2020a. Synchronous speech recognition and speech-to-text translation with interactive decoding. In *Proc. AAAI*, pages 8417–8424.

Yuchen Liu, Jiajun Zhang, Hao Xiong, Long Zhou, Zhongjun He, Hua Wu, Haifeng Wang, and Chengqing Zong. 2020b. Synchronous speech recognition and speech-to-text translation with interactive decoding. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pages 8417–8424.

Zhiyuan Liu, Yankai Lin, and Maosong Sun. 2020c. *Compositional Semantics*, pages 43–57. Springer Singapore, Singapore.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Yizhou Lu, Mingkun Huang, Hao Li, Jiaqi Guo, and Yanmin Qian. 2020. Bi-encoder transformer network for mandarin-english code-switching speech recognition using mixture of experts. In *Proc. Interspeech*.

Yizhou Lu, Mingkun Huang, Xinghua Qu, Pengfei Wei, and Zejun Ma. 2022. Language adaptive cross-lingual speech representation learning with sparse sharing sub-networks. In *ICASSP*.

Loren Lugosch, Tatiana Likhomanenko, Gabriel Synnaeve, and Ronan Collobert. 2022. Pseudo-labeling for massively multilingual speech recognition. In *ICASSP*.

Ne Luo, Dongwei Jiang, Shuaijiang Zhao, Caixia Gong, Wei Zou, and Xiangang Li. 2018. Towards end-to-end code-switching speech recognition. *arXiv preprint arXiv:1810.13091*.

Dau-Cheng Lyu, Eng-Siong Chng, and Haizhou Li. 2013. Language diarization for code-switch conversational speech. In *Proc. ICASSP*.

Dau-Cheng Lyu, Tien-Ping Tan, Eng Siong Chng, and Haizhou Li. 2010. Seame: a mandarin-english code-switching speech corpus in south-east asia. In *Interspeech*.

Tetyana Lyudovyk and Valeriy Pylypenko. 2014. Code-switching speech recognition for closely related languages. In *Proc. SLTU*.

M. Ma, L. Huang, H. Xiong, et al. 2019a. STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proc. ACL*, pages 3025–3036.

Min Ma, Bhuvana Ramabhadran, Jesse Emond, Andrew Rosenberg, and Fadi Biadsy. 2019b. Comparison of data augmentation and adaptation strategies for code-switched automatic speech recognition. In *Proc. ICASSP*.

Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, Alexandre Muzio, Saksham Singhal, Hany Hassan Awadalla, Xia Song, and Furu Wei. 2021. DeltaLM: Encoder-decoder pre-training for language generation and translation by augmenting pretrained multilingual encoders. *arXiv*.

X. Ma, J. Pino, and P. Koehn. 2020a. Simulmt to simulst: Adapting simultaneous text translation to end-to-end simultaneous speech translation. *arXiv preprint arXiv:2011.02048*.

Xutai Ma, Mohammad Javad Dousti, Changhan Wang, Jiatao Gu, and Juan Pino. 2020b. Simuleval: An evaluation toolkit for simultaneous translation. In *Proceedings of the EMNLP*.

Clara Meister, Tim Vieira, and Ryan Cotterell. 2020. Best-first beam search. *Transactions of the Association for Computational Linguistics*, 8:795–809.

Zhong Meng, Naoyuki Kanda, Yashesh Gaur, Sarangarajan Parthasarathy, Eric Sun, Liang Lu, Xie Chen, Jinyu Li, and Yifan Gong. 2021. Internal language model training for domain-adaptive end-to-end speech recognition. In *ICASSP*.

Florian Metze, Thomas Kemp, Thomas Schaaf, Tanja Schultz, and Hagen Soltau. 2000. Confidence measure based language identification. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100)*, volume 3, pages 1827–1830. IEEE.

Bernd T Meyer, Sri Harish Mallidi, Angel Mario Castro Martinez, Guillermo Payá-Vayá, Hendrik Kayser, and Hynek Hermansky. 2016. Performance monitoring for automatic speech recognition in noisy multi-channel environments. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 50–56.

Mehryar Mohri, Fernando Pereira, and Michael Riley. 2002. Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, 16(1):69–88.

Robert C Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 conference short papers*, pages 220–224.

Niko Moritz, Takaaki Hori, and Jonathan Le Roux. 2019. Triggered attention for end-to-end speech recognition. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5666–5670.

Moses-SMT. 2018. `multi-bleu.perl`. `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl`.

Mathias Müller, Annette Rios, and Rico Sennrich. 2020. Domain robustness in neural machine translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas*, pages 151–164, Virtual. Association for Machine Translation in the Americas.

Kenton Murray and David Chiang. 2018a. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, Brussels, Belgium.

Kenton Murray and David Chiang. 2018b. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, Brussels, Belgium. Association for Computational Linguistics.

C. Myers-Scotton. 1993. *Duelling Languages: Grammatical Structure in Codeswitching*. Clarendon Press.

Carol Myers-Scotton. 1997. *Duelling languages: Grammatical structure in codeswitching*. Oxford University Press.

Graham Neubig, Zi-Yi Dou, Junjie Hu, Paul Michel, Danish Pruthi, and Xinyi Wang. 2019. compare-mt: A tool for holistic comparison of language generation systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 35–41, Minneapolis, Minnesota. Association for Computational Linguistics.

H. Ney. 1999. Speech translation: coupling of recognition and translation. In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, volume 1, pages 517–520 vol.1.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual meeting of the Association for Computational Linguistics*, pages 295–302.

ASR Omnilingual, Gil Keren, Artyom Kozhevnikov, Yen Meng, Christophe Ropers, Matthew Setzler, Skyler Wang, Ife Adebara, Michael Auli, Can Balioglu, et al. 2025. Omnilingual asr: Open-source multilingual speech recognition for 1600+ languages. *arXiv preprint arXiv:2511.09690*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *Proc ICASSP*, pages 5206–5210. IEEE.

Sara Papi, Marco Gaido, Matteo Negri, and Marco Turchi. 2022. Over-generation cannot be rewarded: Length-adaptive average lagging for simultaneous speech translation. In *Proceedings of the Third Workshop on Automatic Simultaneous Translation*, pages 12–17.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. Specaugment: simple data augmentation for automatic speech recognition. *Interspeech*.

Vijayaditya Peddinti, Guoguo Chen, Vimal Manohar, Tom Ko, Daniel Povey, and Sanjeev Khudanpur. 2015. JHU ASpIRE system: Robust LVCSR with TDNNS, iVector adaptation and RNN-LMS. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 539–546.

Puyuan Peng, Brian Yan, Shinji Watanabe, and David Harwath. 2023. Prompting the hidden talent of web-scale speech models for zero-shot task generalization. In *Proc. Interspeech 2023*, pages 396–400.

Yifan Peng, Shakeel Muhammad, Yui Sudo, William Chen, Jinchuan Tian, Chyi-Jiunn Lin, and Shinji Watanabe. 2025. OWSM v4: Improving open Whisper-style speech models via data scaling and cleaning. *Proc. Interspeech*, pages 2225–2229.

Yifan Peng, Jinchuan Tian, William Chen, Siddhant Arora, Brian Yan, Yui Sudo, Muhammad Shakeel, Kwanghee Choi, Jiatong Shi, Xuankai Chang, Jee weon Jung, and Shinji Watanabe. 2024. Owsm v3.1: Better and faster open whisper-style speech models based on e-branchformer. In *Interspeech 2024*, pages 352–356.

N. Pham, Thai-Son Nguyen, Thanh-Le Ha, J. Hussain, Felix Schneider, J. Niehues, Sebastian Stüker, and A. Waibel. 2019a. The IWSLT 2019 KIT speech translation system. In *International Workshop on Spoken Language Translation (IWSLT)*.

N.-Q. Pham, T.-S. Nguyen, T.-L. Ha, et al. 2019b. The IWSLT 2019 KIT speech translation system. In *Proc. SIGSLT*.

Alexander Polok, Dominik Klement, Martin Kocour, Jiangyu Han, Federico Landini, Bolaji Yusuf, Matthew Wiesner, Sanjeev Khudanpur, Jan Černockỳ, and Lukáš Burget. 2026. Dicow:: Diarization-conditioned whisper for target speaker automatic speech recognition. *Computer Speech & Language*.

Shana Poplack. 1980. Sometimes i'll start a sentence in Spanish y termino en Español: Toward a typology of code-switching. *The bilingualism reader*, 18(2):221–256.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*.

Matt Post, Gaurav Kumar, Adam Lopez, Damianos Karakos, Chris Callison-Burch, and Sanjeev Khudanpur. 2013. Improved speech-to-text translation with the Fisher and Callhome Spanish–English speech translation corpus. In *International Workshop on Spoken Language Translation (IWSLT 2013)*.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The kaldi speech recognition toolkit. In *Proc. ASRU*.

Rohit Prabhavalkar, Takaaki Hori, Tara N. Sainath, Ralf Schlüter, and Shinji Watanabe. 2024. End-to-end speech recognition: A survey. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:325–351.

Vineel Pratap, Anuroop Sriram, Paden Tomasello, Awni Hannun, Vitaliy Liptchinsky, Gabriel Synnaeve, and Ronan Collobert. 2020a. Massively multilingual asr: 50 languages, 1 model, 1 billion parameters. *Proc. Interspeech*.

Vineel Pratap, Andros Tjandra, Bowen Shi, Paden Tomasello, Arun Babu, Sayani Kundu, Ali Elkahky, Zhaoheng Ni, Apoorv Vyas, Maryam Fazel-Zarandi, et al. 2024. Scaling speech technology to 1,000+ languages. *Journal of Machine Learning Research*, 25(97):1–52.

Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert. 2020b. Mls: A large-scale multilingual dataset for speech research. In *Interspeech 2020*, pages 2757–2761.

Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali. 2018. Language modeling for code-mixing: linguistic theory based synthetic data. In *ACL*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*.

Lihua Qian, Hao Zhou, Yu Bao, Mingxuan Wang, Lin Qiu, Weinan Zhang, Yong Yu, and Lei Li. 2021. Glancing transformer for non-autoregressive neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1993–2003.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *Proc ICML*, pages 28492–28518. PMLR.

SaiKrishna Rallabandi, Sunayana Sitaram, and Alan W Black. 2018. Automatic detection of code-switching style from acoustics. In *Proc. CALCS*.

Q. Ran, Y. Lin, P. Li, and J. Zhou. 2021. Guiding non-autoregressive neural machine translation decoding with reordering information. In *Proc. AAAI*.

Pradeep Rangan, Sundeep Teki, and Hemant Misra. 2020. Exploiting spectral augmentation for code-switched spoken language identification. *arXiv preprint arXiv:2010.07130*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016*.

Vikas Raunak, Siddharth Dalmia, Vivek Gupta, and Florian Metze. 2020. On long-tailed phenomena in neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3088–3095, Online. Association for Computational Linguistics.

Vikas Raunak, Vaibhav Kumar, and Florian Metze. 2019. On compositionality in neural machine translation. *NeurIPS Workshop, Context and Compositionality in Biological and Artificial Neural Systems*.

Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, Ju-Chieh Chou, Sung-Lin Yeh, Szu-Wei Fu, Chien-Feng Liao, Elena Rastorgueva, François Grondin, William Aris, Hwidong Na, Yan Gao, Renato De Mori, and Yoshua Bengio. 2021. SpeechBrain: A general-purpose speech toolkit. ArXiv:2106.04624.

Chandan KA Reddy, Vishak Gopal, and Ross Cutler. 2021. Dnsmos: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6493–6497. IEEE.

Raj Reddy. 1988. Foundations and grand challenges of artificial intelligence: AAAI presidential address. *AI Mag.*, 9(4):9–21.

Shruti Rijhwani, Antonios Anastasopoulos, and Graham Neubig. 2020. OCR post correction for endangered language texts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5931–5942, Online. Association for Computational Linguistics.

Anthony Rousseau, Paul Deléglise, and Yannick Estève. 2012. TED-LIUM: an automatic speech recognition dedicated corpus. In *LREC*.

Paul K Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zalán Borsos, Félix de Chaumont Quitry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, et al. 2023. Audiopalm: A large language model that can speak and listen. *arXiv preprint arXiv:2306.12925*.

Neville Ryant, Kenneth Church, Christopher Cieri, Alejandrina Cristia, Jun Du, Sriram Ganapathy, and Mark Liberman. 2019. The second dihard diarization challenge: Dataset, task, and baselines. *arXiv preprint arXiv:1906.07839*.

Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari. 2022. Utmos: Utokyo-sarulab system for voicemos challenge 2022. In *Interspeech 2022*, pages 4521–4525.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1098–1108, Online. Association for Computational Linguistics.

Tara N Sainath, Ruoming Pang, David Rybach, Yanzhang He, Rohit Prabhavalkar, Wei Li, Mirkó Visontai, Qiao Liang, Trevor Strohman, Yonghui Wu, et al. 2019. Two-pass end-to-end speech recognition. *Proc. Interspeech 2019*, pages 2773–2777.

Haşim Sak, Andrew Senior, Kanishka Rao, Ozan Irsoy, Alex Graves, Françoise Beaufays, and Johan Schalkwyk. 2015. Learning acoustic frame labeling for speech recognition with recurrent neural networks. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4280–4284. IEEE.

Julian Salazar, Davis Liang, Toan Q Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712.

Elizabeth Salesky, Matthias Sperber, and Alan W Black. 2019. Exploring Phoneme-Level Speech Representations for End-to-End Speech Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1835–1841, Florence, Italy. Association for Computational Linguistics.

Elizabeth Salesky, Matthew Wiesner, Jacob Bremerman, Roldano Cattoni, Matteo Negri, Marco Turchi, Douglas W Oard, and Matt Post. 2021. The multilingual TEDx corpus for speech recognition and translation. *arXiv preprint arXiv:2102.01757*.

Lahiru Samarakoon, Brian Mak, and Albert YS Lam. 2018. Domain adaptation of end-to-end speech recognition in low-resource settings. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 382–388. IEEE.

Ramon Sanabria and Florian Metze. 2018. Hierarchical multitask learning with ctc. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 485–490. IEEE.

Seamless Communication. 2023. Seamless: Multilingual expressive and streaming speech translation. In *ArXiv*.

Hiroshi Seki, Takaaki Hori, Shinji Watanabe, Niko Moritz, and Jonathan Le Roux. 2019a. Vectorized beam search for CTC-attention-based speech recognition. In *INTERSPEECH*, pages 3825–3829.

Hiroshi Seki, Takaaki Hori, Shinji Watanabe, Niko Moritz, and Jonathan Le Roux. 2019b. Vectorized beam search for CTC-attention-based speech recognition. In *Proc. Interspeech 2019*, pages 3825–3829.

Hiroshi Seki, Shinji Watanabe, Takaaki Hori, Jonathan Le Roux, and John R Hershey. 2018. End-to-end language-tracking speech recognizer for mixed-language speech. In *ICASSP*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Sanket Shah, Basil Abraham, Sunayana Sitaram, Vikas Joshi, et al. 2020. Learning to recognize code-switched speech without forgetting monolingual speech recognition. *arXiv preprint arXiv:2006.00782*.

Changhao Shan, Chao Weng, Guangsen Wang, Dan Su, Min Luo, Dong Yu, and Lei Xie. 2019a. Component fusion: Learning replaceable language model component for end-to-end speech recognition system. In *Proc. ICASSP*.

Changhao Shan, Chao Weng, Guangsen Wang, Dan Su, Min Luo, Dong Yu, and Lei Xie. 2019b. Investigating end-to-end speech recognition for mandarin-english code-switching. In *ICASSP*.

Jiatong Shi, Dan Berrebbi, William Chen, Ho-Lam Chung, En-Pei Hu, Wei Ping Huang, Xuankai Chang, Shang-Wen Li, Abdelrahman Mohamed, Hung-yi Lee, et al. 2023. Ml-superb: Multilingual speech universal performance benchmark. *Interspeech*.

Xian Shi, Qiangze Feng, and Lei Xie. 2020. The asru 2019 mandarin-english code-switching speech recognition challenge: Open datasets, tracks, methods and results. *Proc. WSTCSMC*.

Sunayana Sitaram, Khyathi Raghavi Chandu, Sai Krishna Rallabandi, and Alan W Black. 2019. A survey of code-switched speech and language processing. *arXiv*.

Sunit Sivasankaran, Brij Mohan Lal Srivastava, Sunayana Sitaram, Kalika Bali, and Monojit Choudhury. 2018. Phone merging for code-switched speech recognition. In *Proc. CALCS*.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*.

Tongtong Song, Qiang Xu, Meng Ge, Longbiao Wang, Hao Shi, Yongjie Lv, Yuqin Lin, and Jianwu Dang. 2022. Language-specific characteristic assistance for code-switching speech recognition. In *Interspeech*.

Xiao Song, Yuexian Zou, Shilei Huang, Shaobin Chen, and Yi Liu. 2017. Investigating multi-task learning for automatic speech recognition with code-switching between mandarin and english. In *Proc. IALP*.

Pavel Sountsov and Sunita Sarawagi. 2016. Length bias in encoder decoder models and a case for global conditioning. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1516–1525, Austin, Texas. Association for Computational Linguistics.

Speechocean. 2017. King-asr-190: Chinese english speech recognition corpus. `http://speechocean.com/`.

M. Sperber, H. Setiawan, et al. 2020. Consistent transcription and translation of speech. *Trans. of the ACL*, 8:695–709.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2019. Attention-passing models for robust and data-efficient end-to-end speech translation. *Transactions of the Association for Computational Linguistics*, 7:313–325.

Matthias Sperber and Matthias Paulik. 2020. Speech translation and the end-to-end promise: Taking stock of where we are. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

F. Stahlberg, D. Saunders, and B. Byrne. 2018. An operation sequence model for explainable neural machine translation. In *Proc. EMNLP*, pages 175–186.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Karan Taneja, Satarupa Guha, Preethi Jyothi, and Basil Abraham. 2019. Exploiting monolingual speech corpora for code-mixed speech recognition. In *Interspeech*.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv*.

Samuel Thomas, Kartik Audhkhasi, and Brian Kingsbury. 2020. Transliteration based data augmentation for training multilingual asr acoustic models in low resource settings. In *Interspeech*.

Jinchuan Tian, Jianwei Yu, Chunlei Zhang, Chao Weng, Yuexian Zou, and Dong Yu. 2022. Lae: Language-aware encoder for monolingual and multilingual asr. In *Interspeech*.

Jörg Tiedemann, Santhosh Thottingal, et al. 2020. Opus-mt–building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*. European Association for Machine Translation.

Christoph Tillmann and Hermann Ney. 2003. Word reordering and a dynamic programming beam search algorithm for statistical machine translation. *Computational linguistics*, 29(1):97–133.

Shubham Toshniwal, Tara N. Sainath, Ron J. Weiss, Bo Li, Pedro Moreno, Eugene Weinstein, and Kanishka Rao. 2018. Multilingual speech recognition with a single end-to-end model. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4904–4908.

Shubham Toshniwal, Hao Tang, Liang Lu, and Karen Livescu. 2017. Multitask learning with low-level auxiliary tasks for encoder-decoder based speech recognition. In *Proc. Interspeech 2017*, pages 3532–3536.

Emiru Tsunoo, Yosuke Kashiwagi, and Shinji Watanabe. 2021. Streaming transformer asr with blockwise synchronous beam search. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 22–29. IEEE.

Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. 2017. Neural machine translation with reconstruction. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 2017*, pages 3097–3103. AAAI Press.

Evelyne Tzoukermann and Corey Miller. 2018. Evaluating automatic speech recognition in translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 2: User Track)*, pages 294–302, Boston, MA. Association for Machine Translation in the Americas.

Enes Yavuz Ugan, Ngoc-Quan Pham, and Alexander Waibel. 2024. DECM: Evaluating bilingual ASR performance on a code-switching/mixing benchmark. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4468–4475, Torino, Italia. ELRA and ICCL.

Jörgen Valk and Tanel Alumäe. 2021. Voxlingua107: a dataset for spoken language recognition. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 652–658. IEEE.

Ewald van der Westhuizen and Thomas Niesler. 2018. A first South African corpus of multilingual code-switched soap opera speech. In *Proc LREC*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, L ukasz Kaiser, and Illia Polosukhin. 2. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ashwin K. Vijayakumar, Michael Cogswell, Ramprasaath R. Selvaraju, Qing Sun, Stefan Lee, David J. Crandall, and Dhruv Batra. 2018. Diverse beam search for improved description of complex scenes. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 7371–7379. AAAI Press.

Alex Waibel. 1996. Interactive translation of conversational speech. *Computer*, 29(7):41–48.

Changhan Wang, Morgane Riviere, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux. 2021. Voxpopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. In *Proc ACL-IJCNLP*.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020a. Fairseq S2T: Fast speech-to-text modeling with fairseq. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (AACL): System Demonstrations*, pages 33–39. Association for Computational Linguistics.

Changhan Wang, Anne Wu, and Juan Pino. 2020b. CoVoST 2: A massively multilingual speech-to-text translation corpus. *arXiv preprint arXiv:2007.10310*.

Chengyi Wang, Yu Wu, Shujie Liu, Zhenglu Yang, and Ming Zhou. 2020c. Bridging the gap between pre-training and fine-tuning for end-to-end speech translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9161–9168.

Qingzheng Wang, Hye-jin Shim, Jiancheng Sun, and Shinji Watanabe. 2025. Geolocation-aware robust spoken language identification. In *Proc. IEEE ASRU*.

Shengye Wang, Li Wan, Yang Yu, and Ignacio Lopez Moreno. 2019. Signal combination for language identification. *arXiv preprint arXiv:1910.09687*.

Shinji Watanabe, Takaaki Hori, and John R. Hershey. 2017a. Language independent end-to-end architecture for joint language identification and speech recognition. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 265–271.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, et al. 2018. Espnet: End-to-end speech processing toolkit. *Proc. Interspeech*.

Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R Hershey, and Tomoki Hayashi. 2017b. Hybrid ctc/attention architecture for end-to-end speech recognition. *JSTSP*.

Shinji Watanabe, Michael Mandel, Jon Barker, Emmanuel Vincent, Ashish Arora, Xuankai Chang, Sanjeev Khudanpur, Vimal Manohar, Daniel Povey, Desh Raj, David Snyder, Aswin Shanmugam Subramanian, Jan Trmal, Bar Ben Yair, Christoph Boeddeker, Zhaoheng Ni, Yusuke Fujita, Shota Horiguchi, Naoyuki Kanda, Takuya Yoshioka, and Neville Ryant. 2020. CHiME-6 Challenge: Tackling Multispeaker Speech Recognition for Unsegmented Recordings. In *6th International Workshop on Speech Processing in Everyday Environments (CHiME 2020)*, pages 1–7.

Jochen Weiner, Ngoc Thang Vu, Dominic Telaar, Florian Metze, Tanja Schultz, Dau-Cheng Lyu, Eng-Siong Chng, and Haizhou Li. 2012. Integration of language identification into a recognition system for spoken conversations containing code-switches. In *Proc. SLTU*.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-sequence models can directly translate foreign speech. In *Proc. Interspeech 2017*, pages 2625–2629.

O. Weller, M. Sperber, C. Gollan, and J. Kluivers. 2021. Streaming models for joint speech recognition and translation. In *Proc. of EACL*, pages 2533–2539.

Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1296–1306, Austin, Texas. Association for Computational Linguistics.

W. Xu and M. Carpuat. 2021. Editor: an edit-based transformer with repositioning for neural machine translation with soft lexical constraints. *Trans. of ACL*, 9:311–328.

Brian Yan, Siddharth Dalmia, Yosuke Higuchi, Graham Neubig, Florian Metze, Alan W Black, and Shinji Watanabe. 2023a. CTC alignments improve autoregressive translation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1615–1631, Dubrovnik, Croatia. Association for Computational Linguistics.

Brian Yan, Siddharth Dalmia, David R. Mortensen, Florian Metze, and Shinji Watanabe. 2021. Differentiable allophone graphs for language-universal speech recognition. In *Interspeech*.

Brian Yan, Patrick Fernandes, Siddharth Dalmia, Jiatong Shi, Yifan Peng, Dan Berrebbi, Xinyi Wang, Graham Neubig, and Shinji Watanabe. 2022a. CMU's IWSLT 2022 dialect speech translation system. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 298–307, Dublin, Ireland (in-person and online). Association for Computational Linguistics.

Brian Yan, Injy Hamed, Shuichiro Shimizu, Vasista Sai Lodagala, William Chen, Olga Iakovenko, Bashar Talafha, Amir Hussein, Alexander Polok, Kalvin Chang, et al. 2025a. CS-FLEURS: A massively multilingual and code-switched speech dataset. In *Proc. Interspeech*, pages 743–747.

Brian Yan, Vineel Pratap, Shinji Watanabe, and Michael Auli. 2025b. Improving multilingual asr in the wild using simple n-best re-ranking. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Brian Yan, Jiatong Shi, Yun Tang, Hirofumi Inaguma, Yifan Peng, Siddharth Dalmia, Peter Polák, Patrick Fernandes, Dan Berrebbi, Tomoki Hayashi, et al. 2023b. Espnet-st-v2: Multipurpose spoken language translation toolkit. *arXiv preprint arXiv:2304.04596*.

Brian Yan, Qingzheng Wang, Matthew Wiesner, Anuj Diwan, Olga Iakovenko, Alex Polok, Injy Hamed, Shuichiro Shimizu, Iris Emerman, Thomas Hain, David R. Mortensen, Peter Viechnicki, and Shinji Watanabe. 2026. Cs-yodas: A mined dataset of in-the-wild code-switched speech. *arXiv preprint arXiv*.

Brian Yan, Matthew Wiesner, Ondřej Klejch, Preethi Jyothi, and Shinji Watanabe. 2023c. Towards zero-shot code-switched speech recognition. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Brian Yan, Chunlei Zhang, Meng Yu, Shi-Xiong Zhang, Siddharth Dalmia, Dan Berrebbi, Chao Weng, Shinji Watanabe, and Dong Yu. 2022b. Joint modeling of code-switched and monolingual asr via conditional factorization. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6412–6416. IEEE.

Brian Yan, Chunlei Zhang, Meng Yu, Shi-Xiong Zhang, Siddharth Dalmia, Dan Berrebbi, Chao Weng, Shinji Watanabe, and Dong Yu. 2022c. Joint modeling of code-switched and monolingual asr via conditional factorization. In *ICASSP*.

Yilin Yang, Liang Huang, and Mingbo Ma. 2018. Breaking the beam search curse: A study of (re-)scoring methods and stopping criteria for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3054–3059, Brussels, Belgium. Association for Computational Linguistics.

Emre Yılmaz, Henk van den Heuvel, and David Van Leeuwen. 2016. Investigating bilingual deep neural networks for automatic recognition of code-switching frisian speech. *Procedia Computer Science*.

Michael Yoder, Shruti Rijhwani, Carolyn Rosé, and Lori Levin. 2017. Code-switching as a social act: The case of arabic wikipedia talk pages. In *NLP+CSS*.

Zhiping Zeng, Yerbolat Khassanov, Van Tung Pham, Haihua Xu, Eng Siong Chng, and Haizhou Li. 2018. On end-to-end mandarin-english code-switching speech recognition. *Interspeech*.

Biao Zhang, Barry Haddow, and Rico Sennrich. 2022a. Revisiting end-to-end speech-to-text translation from scratch. In *International Conference on Machine Learning*.

Chao Zhang, Bo Li, Tara Sainath, Trevor Strohman, Sepand Mavandadi, Shuo-Yiin Chang, and Parisa Haghani. 2022b. Streaming end-to-end multilingual speech recognition with joint language identification. In *Interspeech 2022*, pages 3223–3227.

Shuai Zhang, Jiangyan Yi, Zhengkun Tian, Ye Bai, Jianhua Tao, et al. 2021a. Decoupling pronunciation and language for end-to-end code-switching asr. In *Proc. ICASSP*.

Shuai Zhang, Jiangyan Yi, Zhengkun Tian, Jianhua Tao, and Ye Bai. 2021b. Rnn-t with language bias for end-to-end man-eng code-switching speech recognition. *ISCSLP*.

Weitai Zhang, Zhongyi Ye, Haitao Tang, Xiaoxi Li, Xinyuan Zhou, Jing Yang, Jianwei Cui, Pan Deng, Mohan Shi, Yifan Song, et al. 2022c. The ustc-nelslip offline speech translation systems for iwslt 2022. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 198–207.

Yu Zhang, Wei Han, James Qin, Yongqiang Wang, Ankur Bapna, Zhehuai Chen, Nanxin Chen, Bo Li, and Vera Axelrod et al. 2023. Google usm: Scaling automatic speech recognition beyond 100 languages. *arXiv*.

Chengqi Zhao, Mingxuan Wang, and Lei Li. 2020. NeurST: Neural speech translation toolkit. *arXiv preprint arXiv:2012.10018*.

Jinming Zhao, Vineel Pratap, and Michael Auli. 2024. Scaling a simple approach to zero-shot speech recognition. *arXiv preprint arXiv:2407.17852*.

Miao Zhao, Yufeng Ma, Min Liu, and Minqiang Xu. 2021. The SpeakIn system for VoxCeleb speaker recognition challenge 2021. *arXiv preprint arXiv:2109.01989*.

Renjie Zheng, Junkun Chen, Mingbo Ma, and Liang Huang. 2021. Fused acoustic and text encoding for multimodal bilingual pretraining and speech translation. *arXiv preprint arXiv:2102.05766*.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2019. Understanding knowledge distillation in non-autoregressive machine translation. In *International Conference on Learning Representations*.

Long Zhou, Jinyu Li, Eric Sun, and Shujie Liu. 2022a. A configurable multilingual model is all you need to recognize all languages. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6422–6426. IEEE.

Wei Zhou, Wilfried Michel, Kazuki Irie, Markus Kitza, Ralf Schlüter, and Hermann Ney. 2020a. The rwth asr system for ted-lium release 2: Improving hybrid hmm with specaugment. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7839–7843. IEEE.

Wei Zhou, Zuoyun Zheng, Ralf Schlüter, and Hermann Ney. 2022b. On language model integration for rnn transducer based speech recognition. In *ICASSP*.

Xinyuan Zhou, Emre Yılmaz, Yanhua Long, Yijie Li, and Haizhou Li. 2020b. Multi-encoder-decoder transformer for code-switching speech recognition. *Interspeech*.